TSC 6800 Mnemonic Assembler

SL68-26

TECHNICAL SYSTEMS CONSULTANTS

TSC 6800 Mnemonic Assembler

COPYRIGHT © 1978 BY
Technical Systems Consultants, Inc.
P.O. Box 2574
West Lafayette, Indiana 47906
All Rights Reserved

Copyright Notice

This entire manual, source listing and documentation is provided for personal use and enjoyment by the purchaser The entire contents have been copyrighted by Technical Systems Consultants, Inc., and reproduction by any means is prohibited. Use of this program, or any part thereof, for any purpose other than single end use is strictly prohibited.

TSC 6800 Mnemonic Assembler
SL68-26
Copyright (C) 1977
Technical Systems Consultants
Box 2574
West Lafayette, IN 47906

The TSC 6800 Mnemonic Assembler was written for maximum flexibility making it usable to owners of RAM-only systems as well as disk system owners. As always, flexibility adds complexity and therefore the user is advised to read the following application notes thoroughly before trying to use this program.

It is assumed that the user is familiar with assembly language and, in particular, the mnemonics of the M68ØØ assembly language. Those who are not are referred to the "M68ØØ Microprocessor Programming Manual" or the "M68ØØ Programming Reference Manual," both available from your Motorola distributor.

The source language (input) for the TSC 68ØØ Mnemonic Assembler consists of a subset of the 7-bit ASCII (American Standard Code for Information Interchange, 1968) character set. Special meaning is attached to many of these characters as will be described later. In all cases the parity bit (most significant bit) of each character must be Ø. This restriction, of course, does not apply to line numbers, if present.

Each line of source for the assembler consists of any number of bytes (possibly none) preceeding the first character of the source statement, followed by the source statement, followed by a carriage return (hex ØD). The source statement consists of up to four "fields" which are free format. From left to right, the four fields are label , operator (mnemonic),

operand, and comment. There must be at least one space between each of these fields. Further restrictions and options for each of these fields are:

label field

1) The label must begin in the first column and must be unique.
2) Labels consist of letters (A-Z) and numerals (Ø-9).
3) Every label must begin with a letter (A-Z).
4) Only the first 6 characters of any label are significant, the rest are ignored.
5) The label field may be the only field present.

operator field

1) The operator is 3 alphabetic characters (A-Z) which must be followed by a space. The exception to this is number 2, below.
2) Mnemonics such as LDA A and AND B may be written as LDAA and ANDB, respectively. In this case fourth character must be followed by a space.

operand field

1) The operand field may consist of an addressing mode indicator and an expression or just an expression.
2) The addressing mode indicator is either a # (Pound sign) followed by an expression for immediate addression or an expression followed by ,X for indexed addression. (Expressions defined later.)
3) An operand may or may not be required depending on the addressing mode.

comment field

1) The comment field is optional

2) Comments may contain any character from SPACE ($2Ø) to DEL ($7F).

Expressions

Expressions consist of combinations of numbers and symbols seperated by one of the four arithmetic operators +, - , *, /. The arithmetic is done with 16 bit integer operands and truncated as necessary. 8 bit results are taken from the least significant 8 bits. Unary (+) and (-) are allowed. Expressions must not contain spaces.

Numbers

Numbers are groupings of the numerals Ø-9 and possibly letters prefixed or postfixed by a base indicator. Possible base indicators are shown below. The ASCII base allows a single ASCII character ($2Ø-$5F) to be used as an operand when preceeded by a single quote.

Base	Prefix	Postfix	Comment
Decimal	none	none	decimal assumed
Binary	%	B	Ø, 1 allowed
Octal	@	O or Q	Ø-7 allowed
Hexadecimal	$	H	Ø-9, A-F allowed
ASCII	'	not allowed	ASCII equivalence

Symbols

Symbols are groupings of letters and numerals the first 6 of which are significant and the first of which must be a letter. The single character * is a special symbol whose value is the current value of the program counter (PC).

Evaluation of Symbols and Expressions

Since this is a two pass assembler all symbols must be resolved in the two passes. Therefore, only one level of forward referencing is allowed.

Assembler Directives

In addition to the 72 M68ØØ mnemonics this assembler supports 11 assembler directives or pseudo-ops. These pseudo-ops are listed below along with a brief description. More detailed descriptions follow.

FCC	form constant character
FCB	form constant byte
FDB	form double byte
SPC	insert spaces in output listing
OPT	activate or deactivate assembler options
PAG	skip to next page of output
ORG	define new origin (PC)
EQU	assign value to symbol
END, MON	signal end of source program
NAM, TTL	specify name or title
RMB	reserve memory bytes

FCC

The function of FCC is to create character strings for messages or tables. The character string 'text' is broken down to ASCII, one character per byte. The two allowable formats are shown below:

```
label     FCC     count, text
```

or

```
label     FCC     delimiter text same delimiter
```

where count is any legal expression. In the case where a number is used as a delimiter the first character of text must not be a comma. The character limit of any single FCC statement is 255. The use of label is optional.

FCB

The FCB pseudo-op causes an expression to be evaluated and the resultant 8 bits placed in memory. Usage is shown below:

```
label     FCB     expression 1, expression 2,...,expression N
```

Each expression is seperated by a comma with a maximum of 255 expressions per FCB statement. The label is optional.

FDB

The function of the FDB directive is identical to FCB except 16 bit quantities are assembled, i.e., two bytes generated for each expression. The required format is shown below:

```
label     FDB     expression 1, expression 2,...,expression N
```

where the label is optional. The maximum number of expressions is 127.

SPC

The SPC operator causes the specified number of spaces to be inserted in the output listing. The format is shown below.

```
          SPC     expression
```

Notice that no label is allowed. If 'expression' evaluates to zero one space is inserted. The operator SPC itself does not appear in the output listing. If PAGE mode is selected SPC will not cause spacing past the top of the next page.

OPT

The directive OPT is used to activate or deactivate the assembler options. The format is shown below. Notice that no label is allowed and no code is generated.

```
OPT     option 1, option 2,...,option N
```

The allowable options are:

SYM	print sorted symbol table after the listing (default)
NOS	do not print the symbol table
GEN	print all code generated by FCB, FDB, and FCC (default)
NOG	print only one line for each FCB, FDB, or FCC
LIS	print the assembled source listing (default)
NOL	suppress the printing of the source listing
PAG	enable page formatting and numbering
NOP	disable page mode (default)
MEM	enable storing of object code in memory
NOM	disable storing of object code in memory (default)
TAP	enable the production of MIKBUG object tape
NOT	disable the production of MIKBUG object tape (default)

If contradicting options appear the last one appearing takes precedence. All options take effect simultaneously at the beginning of pass 2. The default options specified take effect unless the user specifies a particular option. Only the first 3 characters of an option name are significant and multiple options are seperated by a comma. Some of the consequences and uses of the options will be explained later.

PAG

The PAG operator, if the PAG option is on, causes a page eject and subsequently causes the title (if any) and page number to be printed at the top of the next page. No label is allowed and no code is produced. Notice that the first page of any listing is page Ø and no title is printed on that page. The PAG operator itself will not appear in the listing.

The usual procedure is to have all the options and the title declaration followed by a PAG be the first statements in a program.

ORG

The ORG operator, whose format is shown below, causes a new origin address (PC) for the code following.

```
        ORG     expression
```

No label is allowed and no code is produced. If no ORG appears an origin of ØØØØ is assumed.

EQU

EQU is used to equate a symbol to an expression as shown below. A label is required and no code is generated. Only one level of forward referencing is allowed and the equate must not be recursive.

```
label   EQU     expression
```

No code is produced by EQU.

END or MON

These operators signal the assembler that the end of the source input has occurred. No label is allowed and no code is generated.

NAM or TTL

These operators are used to assign a title to be printed at the top of all pages (other than page Ø) if the PAG option is on. If the PAG option is off this operator has no effect. The format, as shown below allows up to 32 characters in the title. No label is allowed

```
TTL     text for the title
```

and no code is generated. If more than one TTL or NAM operator appears the last one "executed" will be printed on the next page.

RMB

This operator causes the assembler to reserve memory for data storage. No code is produced and therefore the contents of those memory locations are undefined at run time. The label is optional as shown below

```
label     RMB     expression
```

where 'expression' is a 16 bit quantity.

** Description of assembler operation

Pass 1 - PASONE ($03B1)

Pass 1 is used to build the symbol table which is used to resolve forward references. Nothing is printed unless the error limit is exceeded (85). Pass 1 must be run before PASS 2 and again before PASS 3.

Pass 2 - PASTWO ($03D9)

During pass 2 several things may happen.

1) If the LIST option is on, the assembled source listing is printed with error messages, if any.

2) If the LIST option is off only offending source lines and their corresponding error messages are printed.

3) If the TAPE option is on, a MIKBUG formatted object record is outputted (through a different control point than the source listing).

4) If the MEMORY option is on, object code is placed in memory in the following form:

COUNT ADDRESS DATA ... DATA COUNT ADDRESS DATA ... DATA TERM

where ADDRESS is the destination address of the first data following

COUNT is a 16 bit byte count indicating how many data bytes follow

DATA is the actual data

TERM is the record terminator (a COUNT of ØØØØ)

When a count of ØØØØ occurs this signifies the end of the program.

5) If the SYMBOL option is on, a sorted symbol table will be printed after the assembly listing (if any). Pass 1 must be run before PASS 2.

Pass 3 - PASTHR ($05BB)

Pass 3 is used when the user does not have a "punch" device, on which to save the MIKBUG formatted records, which operates independently from the list device. Pass 3 is identical in operation to pass 2 except that NOSYM, NOLIST, NOMEM and TAPE options are forced and error messages are suppressed. Pass 1 must be run before PASS 3, PASS 2 and PASS 3 are independent.

Initialization

There exists in the assembler an initialization routine for each of the passes which must be run once before that pass in run. These are called P1INIT, P2INIT, and P3INIT for passes 1, 2, and 3, respectively.

Adapting to Your System

Due to the inherent flexibility of this assembler it is necessary that each user customize it to fit the particular system. This involves very few changes and can be made by any individual familiar with 6800 assembly language. Each point to be adapted is explained below.

Output Character Routine

The address at $0321 must be changed to that of your Output Character routine. This routine must print the ASCII character in the A register whose parity bit (most significant bit) is zero. The B and X registers must not be altered. If you have a printer or a disk you will likely want to specify the address of a program which handles these peripherals as well as the control terminal.

Tape Output Character Routine

The address at $0324 must be changed to that of your tape punch (or tape record) routine. It is through this control point that the MIKBUG formatted object code is outputted. If you do not have a seperate punch or record device this address may be the same as the Output Character routine address, i.e., tape device same as list device.

Tape Control Characters

There are provisions at $04C0 and $04C4 for four control characters to activate and deactivate, respectively, your punch or record device. Simply place the appropriate control characters for your device in each of the strings. If you desire to send less than the four characters, change the byte at $04B3 to the appropriate value (even Ø). This will, of course, affect both turn on and turn off simultaneously.

Tape Control Delay

The byte at $04C9 controls the number of half-seconds (1MHz clock) of delay between tape turn on and data and also between data and tape turn off. The delay is set now to 2 seconds. If you don't need delay at all set the byte to ØØ.

Page Control

Page Eject

The four bytes at $11D1 are provided for the user to insert the necessary control characters to cause the printer to form feed, i.e., eject to the top of the next page. If you need only 1 character, simply place the Ø4 after that character in the string. The control character is currently set to $ØA (line feed).

Top Margin Control

The byte at $1143 controls the number of lines from the form feed position to the title and page number line (can be Ø).

Page Length Control

The byte at $07C5 controls the number of lines to be printed on each page before the form feed is issued. This count includes the top margin and the title line and should be larger than (top margin + 1).

The user may want to alter other features such as the number of columns printed in the symbol table, etc. Most modifications of this type will be needed by only a few users and therefore will not be elaborated upon here. These users are encouraged to study the code to facilitate making the desired modifications.

Controller Routine

The routine MAIN ($3ØØ) is an example of how to use the assembler subroutines. It assumes the user has no independent punch device and therefore must run PASS 3 in order to output the object code. Also, MAIN assumes the source program resides entirely in RAM and that the necessary pointers (to be described) are set.

Disk users will, of course, want to write their own MAIN routine which will bring in each section of source code and run PASS 1 on each, then bring in each section again and run PASS 2, similarly for PASS 3. Naturally, the initiatization routine for each pass need be run only once before each series of passes of the same type. Be reminded that PASS 1 needs to be run before PASS 2 and again before PASS 3. This procedure will allow assembly of files too large to reside entirely in RAM.

One note of caution: the END operator is not strictly necessary at the end of a program as the pass in effect will terminate at the end of the source area. However, if you are generating object code, only an END statement will flush the code buffer or fix the memory count. Likewise, only an END operator will cause the symbol table to be printed (if SYM is on). The byte ENDFLG ($0058) will be set ($FF) if the END operator occurs, which can be detected by your MAIN routine.

Assembler Data Pointers

Before calling any assembler routines the user must set several pointers to data areas. This feature allows much flexibility but restrictions which apply to each pointer are outlined below. No assembler routines modify these pointers.

LBLBEG - $ØØ4Ø

LBLEND - $ØØ42

These are the pointers to the area which will be used for the label table (symbol table). Each entry (symbol) in the table requires 8 bytes. A large table will result in the Put Label and Find Label routines running faster but the Shell (sort) routine will run slower. A small table will have the opposite effect. Of course, the table needs to be large enough to accomodate the number of symbols in your program. A reasonable formula for determining the size necessary is:

SIZE = N * 8 * 2 = N * 16 bytes

where N is an estimate of the number of symbols expected. When the table is full an error message will be inserted in the listing. (The table may not be completely full due to the algorithm used for creating the table - hashing, or scatter storage.)

If you want a 1K symbol table (a recommended minimum, enough for 60-80 labels) you might set LBLBEG to $2ØØØ and LBLEND to $23FF. Notice that the pointers do point to the actual beginning and end of the table.

SRCBEG - $ØØ44

SRCEND - $ØØ46

These two pointers indicate the beginning and end of the section of source code to be assembled, which may be as small as one line of source. SRCEND must point to the carriage return ($ØD) of the last line of the source section to be assembled.

LINBYT - $ØØ48

Although not actually a pointer LINBYT is related to the source pointers. It tells the assembler how many bytes to ignore from carriage return of the previous line (or SRCBEG) before actually processing text. This allows direct output of text editors to be assembled without removing the preceeding line numbers. If you have no line number bytes, set LINBYT to Ø.

MEMPTR - $ØØ49

This pointer tells the assembler where in memory, if the MEMORY option is on, to put the assembled object code. Recall that four extra bytes (address and count) are required for each contiguous block of code.

Error Messages

This assembler supports 12 error messages which are printed after the offending line. The error messages announce violations of any of the restrictions set forth in this manual and are, therefore, self-explanatory.

Additionally, the byte 'ERRORS' (cleared by PIINIT) will be set if any errors have occurred in any of the passes.

Note: The ASCII characters ØØ - $ØC, $ØE - $1F, and $8Ø - $8F, inclusively are explicitly prohibited from being in any area of the source program with the exception of the bytes which are skipped by the assembler (line number bytes). Their existence will cause undefined results. The remaining ASCII characters may appear subject to all of the foregoing restrictions.

Additional Feature

This assembler supports 2 extra mnemonics namely BHS and BLO which are the logical equivalents of BCC and BCS respectively. However, Branch if Higher or Same and Branch if Lower are much easier to remember and use.

Final Note

Please be reminded, when using the MEMORY option, that in most cases the object code will not be put in memory where it can be executed. It is up to the user to write the simple routine necessary to move the code to its proper executable location.

Important: The address at $031C is the address to which control returns when the assembly is complete. This should be modified to suit your needs.

**** USING THE TSC EDITOR ****

The TSC Text Editing System and the TSC Mnemonic Assembler have not been written to be used co-resident. It is possible to use them one after the other without reloading the source. Following is the procedure to be used:

1. Load the editor but before running, change BEGPNT (location $0359) presently $1492 to $1600. This moves the starting location of the text. Put a $ØD at location $15FF.
2. Run the editor and create your file.
3. When finished, exit the editor and write down the contents of
   a.) FILBEG ($0097- 0098) Shows the source beginning.
   b.) FILEND ($0099-009A) Shows one past the source end.
4. Load the assembler but before running be sure to set all pointers.
   a.) Symbol Table limits ($0040-0043)
   b.) Source beginning ($0044-0045) contents of edit FILBEG
   c.) Source ending ($0046-0047) "contents-1" of edit FILEND
       ***** Be sure to subtract one from FILEND !!
   d.) Skip count ($0048) Set this to 03 (3 line no. in editor)
   e.) Memory pointer ($0049) Set if used.
5. Run the assembler.

```
LOCN B1 B2 B3
                    *
                    *
                    * TSC 6800 ASSEMBLER SYSTEM
                    *  COPYRIGHT 1977 (C) BY
                    *
                    * TECHNICAL SYSTEMS CONSULTANTS, INC.
                    * PO BOX 2574
                    * WEST LAFAYETTE, INDIANA 47906
                    *
                    *



                    *
                    * INSTRUCTION TYPES
                    *  TYPE 1     INHERENT
                    *  TYPE 2     RELATIVE
                    *  TYPE 3     INDEXED,EXTENDED    0,1
                    *  TYPE 4     DIRECT,INDEXED,EXTENDED    0,1,2
                    *  TYPE 5     IMMEDIATE,DIRECT,INDEXED,EXTENDED  0,1,2,3
                    *  TYPE 6     INHERENT (A,B),INDEXED,EXTENDED 0,1,2,3
                    *  TYPE 7     INHERENT (A,B)  0,1
                    *  TYPE 8     FCC
                    *  TYPE 9     FCB
                    *  TYPE 10    FDB
                    *  TYPE 11    SPC
                    *  TYPE 12    OPT
                    *  TYPE 13    PAG
                    *  TYPE 14    ORG
                    *  TYPE 15    EQU
                    *  TYPE 16    END,MON
                    *  TYPE 17    NAM,TTL
                    *  TYPE 18    RMB
                    *
                    *
                    *  ERROR TYPES
                    *
                    *  0    SYMBOL TABLE FULL
                    *  1    UNDEFINED SYMBOL
                    *  2    MULTIPLY DEFINED SYMBOL
                    *  3    UNRECOGNIZABLE MNEMONIC
                    *  4    ILLEGAL CHARACTER IN LABEL
                    *  5    ILLEGAL CHARACTER IN OPERAND
                    *  6    RELATIVE BRANCH TOO LONG
                    *  7    SYNTAX ERROR
                    *  8    ILLEGAL INDEX VARIABLE
                    *  9    ILLEGAL CHARACTER FOR SPECIFIED BASE
                    * 10    ILLEGAL OPTION SWITCH
                    * 11    TOO MANY OPERANDS IN DATA STATEMENT
                    *
                    *
                    *  STORAGE
                          ORG     $40
                    *
0040                LBLBEG  RMB     2
0042                LBLEND  RMB     2
```

```
LOCN B1 B2 B3
0044            SRCBEG  RMB     2
0046            SRCEND  RMB     2
0048            LINBYT  RMB     1
0049            MEMOBJ  RMB     2
004B            PC      RMB     2
004D            SRCPTR  RMB     2
004F            LABEL   RMB     6
0055            PRFLG   RMB     1
0056            ERRFLG  RMB     1
0057            MATFLG  RMB     1
0058            ENDFLG  RMB     1
0059            PCFLAG  RMB     1
005A            DATFLG  RMB     1
005B            FCCFLG  RMB     1
005C            EJFLG   RMB     1
005D            P3FLG   RMB     1
005E            PRTFLG  RMB     1
005F            PAGFLG  RMB     1
0060            LBLMSK  RMB     1
0061            CKSUM   RMB     1
0062            OBJINT  RMB     1
0063            OPN     RMB     1
0064            TERM    RMB     1
0065            XSAVE   RMB     2
0067            SPSAVE  RMB     2
0069            XTEMP   RMB     2
006B            XTEMP1  RMB     2
006D            XTEMP2  RMB     2
006F            XTEMP3  RMB     2
0071            XTEMP4  RMB     2
0073            XTEMP5  RMB     2
0075            LTEMP   RMB     2
0077            QTEMP3  RMB     2
0079            QTEMP2  RMB     2
007B            QTEMP   RMB     2
007D            TEMP    RMB     1
007E            OPCODE  RMB     1
007F            OP1     RMB     1
0080            OP2     RMB     1
0081            P2ERR1  RMB     1
0082            P2ERR2  RMB     1
0083            P2ERR3  RMB     1
0084            LSTERR  RMB     1
0085            ERRPTR  RMB     2
0087            BYTPTR  RMB     2
0089            OBJPTR  RMB     2
008B            MEMPTR  RMB     2
008D            LINPTR  RMB     2
008F            PASS    RMB     1
0090            OPCNT   RMB     1
0091            RNDM    RMB     3
0094            OPTPTR  RMB     2
0096            OPNPTR  RMB     2
0098            SAVPTR  RMB     2
009A            MCOUNT  RMB     2
```

```
LOCN B1 B2 B3
009C             LSTPCM  RMB     2
009E             LASTPC  RMB     2
00A0             OBJADR  RMB     2
00A2             LASTM   RMB     2
00A4             HASHCT  RMB     1
00A5             ERRCNT  RMB     1
00A6             BYTCNT  RMB     1
00A7             BUFCNT  RMB     1
00A8             LINCNT  RMB     1
00A9             ERRORS  RMB     1
00AA             GAP     RMB     1
00AB             MODFY   RMB     1
00AC             PAGENO  RMB     2
00AE             LIST    RMB     1
00AF             SYMBOL  RMB     1
00B0             GENER   RMB     1
00B1             PAGER   RMB     1
00B2             TAPE    RMB     1
00B3             MEMORY  RMB     1
00B4             OBJBUF  RMB     18
00C6             TITLE   RMB     33
                 *
                 *
                 *
      0036       LINES   EQU     54
      000A       EJCHR   EQU     $0A
                 *
                 *
                         ORG     $100
0100             ERRSTK  RMB     256
0200             BYTSTK  RMB     256
                 *
                 *
0300 8E A0 7F    MAIN    LDS     #$A07F     SET STACK ************
0303 BD 03 26            JSR     P1INIT
0306 BD 03 B1            JSR     PASONE
0309 BD 03 6F            JSR     P2INIT
030C BD 03 D9            JSR     PASTWO
030F BD 03 26            JSR     P1INIT
0312 BD 03 B1            JSR     PASONE
0315 BD 03 6F            JSR     P3INIT
0318 BD 05 BB            JSR     PASTHR
                 *
                 * EXTERNAL LINKAGES
031B 7E E0 D0    MON     JMP     $E0D0        RETURN TO MONITOR  PROGRAM
031E 86 20       OUTS    LDA A   #'
0320 7E E1 D1    OUTCH   JMP     $E1D1
0323 7E E1 D1    TAPOUT  JMP     $E1D1
                 *
                 *
                 *
                 ** P1INIT
                 * PASS 1 INITIALIZATION.  MUST BE
                 * RUN BEFORE A SERIES OF PASS 1 RUNS.
0326 86 FF       P1INIT  LDA A   #$FF
```

```
LOCN B1 B2 B3
0328 97 AE              STA A  LIST
032A 97 B0              STA A  GENER
032C 97 AF              STA A  SYMBOL
032E 97 59              STA A  PCFLAG
0330 40                 NEG A
0331 97 A8              STA A  LINCNT     INITIALIZE COUNT
0333 4F                 CLR A
0334 97 B1              STA A  PAGER      SET 'OFF' OPTIONS
0336 97 AC              STA A  PAGENO
0338 97 AD              STA A  PAGENO+1
033A 97 A5              STA A  ERRCNT     SET COUNT
033C 97 56              STA A  ERRFLG     CLEAR FLAG
033E 97 B2              STA A  TAPE
0340 97 B3              STA A  MEMORY
0342 97 58              STA A  ENDFLG     CLR FLAG
0344 97 A9              STA A  ERRORS
0346 86 7F              LDA A  #$7F
0348 97 60              STA A  LBLMSK     SET MASK
034A CE 01 00           LDX    #ERRSTK
034D DF 85              STX    ERRPTR     SET POINTER
034F DE 40              LDX    LBLBEG     GET LABEL TABLE START
0351 6F 00      CLRLBL  CLR    0,X        SET WHOLE TABLE TO 0
0353 08                 INX
0354 9C 42              CPX    LBLEND     CHECK DONE
0356 26 F9              BNE    CLRLBL     LOOP TILL DONE
0358 CE 00 C6           LDX    #TITLE
035B 86 20              LDA A  #'
035D A7 00      SETTL   STA A  0,X
035F 08                 INX
0360 8C 00 E6           CPX    #TITLE+32  CHECK ALL DONE
0363 26 F8              BNE    SETTL      GO FINISH
0365 86 04              LDA A  #4
0367 A7 00              STA A  0,X        SET EOT
0369 CE 00 00           LDX    #0
036C DF 4B              STX    PC         SET PC TO 0
036E 39                 RTS
                *
                *
                ** P2INIT
                * PASS 2 INITIALIZATION.  MUST BE RUN
                * BEFORE A SERIES OF PASS 2 RUNS.
036F 86 FF      P2INIT  LDA A  #$FF
0371 97 62              STA A  OBJINT     SET TOGGLE
0373 97 5D              STA A  P3FLG      SET NOT PASS 3
0375 CE 01 00           LDX    #ERRSTK
0378 DF 85              STX    ERRPTR     INITIALIZE ERROR PTR
037A CE 00 00           LDX    #0
037D DF 4B              STX    PC         INITIALIZE PC
037F CE FF FF           LDX    #$FFFF
0382 DF 9C              STX    LSTPCM
0384 DF 9E              STX    LASTPC     SET OBJECT PC'S
0386 4F                 CLR A
0387 97 A7              STA A  BUFCNT
0389 97 9A              STA A  MCOUNT
038B 97 9B              STA A  MCOUNT+1
```

```
LOCN B1 B2 B3
038D 97 58              STA A  ENDFLG     CLEAR FLAG
038F CE 00 B4           LDX    #OBJBUF
0392 DF 89              STX    OBJPTR     SET OBJECT PTR
0394 DE 49              LDX    MEMOBJ
0396 DF 8B              STX    MEMPTR     SET  MEMORY PTR
0398 DF A2              STX    LASTM
039A DE 40              LDX    LBLBEG     GET LABEL PTR
039C A6 00      SETBIT  LDA A  0,X        GET FIRST CHAR
039E 27 04              BEQ    NOLAB      IF 0, NO LABEL
03A0 8A 80              ORA A  #$80       SET FLAG BIT
03A2 A7 00              STA A  0,X        PUT BACK
03A4 C6 08      NOLAB   LDA B  #8         SET COUNT
03A6 08         ADVPTR  INX               MOVE PTR
03A7 9C 42              CPX    LBLEND     SEE IF DONE
03A9 27 05              BEQ    P2IN3
03AB 5A                 DEC B
03AC 26 F8              BNE    ADVPTR     SEE IF AT NEW POSITION
03AE 20 EC              BRA    SETBIT     GO SET NEXT FLAG
03B0 39         P2IN3   RTS
                *
                *
                ** P3INIT
                * PASS 3 INITIALIZATION
      036F      P3INIT  EQU    P2INIT     SAME AS PASS 2
                *
                *
                ** PASONE
                * PERFORMS ASSEMBLY PASS 1
03B1 9F 67      PASONE  STS    SPSAVE     SAVE SP
03B3 DE 44              LDX    SRCBEG     GET SOURCE POINTER
03B5 09                 DEX               ADJUST
03B6 7F 00 8F           CLR    PASS       SET PASS1
03B9 DF 4D      PASS1   STX    SRCPTR     SAVE PTR
03BB BD 0B 75           JSR    PARSE      PARSE UP THE LINE
03BE DF 6F              STX    XTEMP3     SAVE SOURCE POINTER
03C0 96 4F              LDA A  LABEL      GET FIRST CHAR OF LAB.
03C2 27 03              BEQ    PASS11     IF NO LABEL
03C4 BD 08 A2           JSR    PUTLBL     GO INSTALL LABEL
03C7 96 55      PASS11  LDA A  PRFLG      GET PROCESS FLAG
03C9 26 03              BNE    PASS12     IF SET, PROCESS
03CB BD 0C 44           JSR    FND222     GO GET OPERATOR
03CE DE 6F      PASS12  LDX    XTEMP3     GET SOURCE PTR
03D0 96 58              LDA A  ENDFLG
03D2 26 04              BNE    PASS13
03D4 9C 46              CPX    SRCEND     CHECK DNE
03D6 26 E1              BNE    PASS1      IF NOT, LOOP
03D8 39         PASS13  RTS
                *
                *
                *
                ** PASTWO
                * PERFORMS ASSEMBLY PASS 2
03D9 DE 44      PASTWO  LDX    SRCBEG     POINT TO BEGIN. SOURCE
03DB 09                 DEX               ADJUST
03DC 86 01              LDA A  #$01
```

```
LOCN B1 B2 B3
03DE 97 8F            STA A  PASS      SET PASS 2
03E0 DF 4D     PASS2  STX    SRCPTR    SAVE POINTER
03E2 DE 4B            LDX    PC
03E4 DF 6D            STX    XTEMP2    SAVE PC
03E6 DE 4D            LDX    SRCPTR    GET POINTER
03E8 BD 0B 75  PASS2A JSR    PARSE     GO PARSE THE LINE
03EB DF 6F            STX    XTEMP3    SAVE PTR
03ED 96 4F            LDA A  LABEL     GET FIRST CHAR
03EF 27 09            BEQ    PASS2B    IF NOT THERE, SKIP
03F1 BD 09 05         JSR    FNDLBL    LOCATE LABEL
03F4 A6 00            LDA A  0,X       GET FIRST CHAR
03F6 84 7F            AND A  #$7F      RESET BIT
03F8 A7 00            STA A  0,X       PUT BACK
03FA 96 55     PASS2B LDA A  PRFLG     GET PROCESS FLAG
03FC 26 03            BNE    PASS2X    IF SET, DONT PROCESS
03FE BD 09 1F         JSR    FNDOPT    GET OPERATION
0401 96 90     PASS2X LDA A  OPCNT     CHECK BYTE COUNT
0403 27 16            BEQ    PASS2C    IF 0, SKIP
0405 96 5D            LDA A  P3FLG     CHECK PASS 3
0407 27 04            BEQ    OBJGEN    IF SO, GO GENERATE CODE
0409 96 B2            LDA A  TAPE      SEE IF TAPE ON
040B 27 07            BEQ    MEMGEN    IF NOT, CHECK MEMORY
040D BD 14 89  OBJGEN JSR    OBJCOD    GO GENERATE CODE
0410 96 5D            LDA A  P3FLG     CHECK PASS3
0412 27 07            BEQ    PASS2C    IF SO, SKIP MEMORY
0414 96 B3     MEMGEN LDA A  MEMORY    SEE IF MEMORY ON
0416 27 03            BEQ    PASS2C    IF NOT, SKIP
0418 BD 15 77         JSR    MEMCOD    GO PUT IN MEMORY
041B 96 5D     PASS2C LDA A  P3FLG     CHECK PASS3
041D 26 03            BNE    SHORT
041F 7E 04 A4         JMP    NOERR4
0422 96 5E     SHORT  LDA A  PRTFLG    SEE IF PRINT
0424 27 0D            BEQ    CHK2ER    IF NOT, SKIP
0426 96 AE            LDA A  LIST      GET LIST FLAG
0428 27 09            BEQ    CHK2ER    SKIP IF NO LIST
042A 96 90            LDA A  OPCNT
042C 36               PSH A
042D BD 05 C1         JSR    PRTINF    GO PRINT DATA
0430 32               PUL A
0431 97 90            STA A  OPCNT      RESTORE COUNT
0433 86 FF     CHK2ER LDA A  #$FF
0435 97 56            STA A  ERRFLG     SET FLAG
0437 96 A5     CHKERR LDA A  ERRCNT     GET COUNT
0439 27 3A            BEQ    NOERR     IF 0, NO ERRORS
043B DE 85            LDX    ERRPTR    GET POINTER
043D EE 00            LDX    0,X       GET ERR ADDRESS
043F 9C 4D            CPX    SRCPTR    CHECK IF HERE
0441 26 32            BNE    NOERR     IF NOT, NO ERROR
0443 96 AE            LDA A  LIST      GET LIST FLAG
0445 26 06            BNE    GETERR    IF LIST ON, SOURCE PRINTED
0447 BD 05 FF         JSR    PRTDAT    PRINT DATA
044A BD 06 42         JSR    PRTSRC  GO PRINT SOURCE TOO
044D DE 85     GETERR LDX    ERRPTR    GET ERROR PTR
044F 7A 00 A5         DEC    ERRCNT    COUNT ONE DOWN
0452 E6 02            LDA B  2,X       GET TYPE
```

```
LOCN B1 B2 B3
0454 27 15          BEQ     GETER2
0456 D1 81          CMP B   P2ERR1      CHECK SAME
0458 26 03          BNE     CHK2
045A 7F 00 81       CLR     P2ERR1
045D D1 82    CHK2  CMP B   P2ERR2
045F 26 03          BNE     CHK3
0461 7F 00 82       CLR     P2ERR2
0464 D1 83    CHK3  CMP B   P2ERR3
0466 26 03          BNE     GETER2
0468 7F 00 83       CLR     P2ERR3
046B 08       GETER2 INX
046C 08              INX
046D 08              INX
046E DF 85           STX     ERRPTR      STORE NEW PTR
0470 BD 06 51        JSR     PRTERR      GO INSERT ERROR MESSAGE
0473 20 C2           BRA     CHKERR      GO SEE IF MORE ERRORS
0475 CE 00 81 NOERR  LDX     #P2ERR1     POINT TO STORE
0478 86 03           LDA A   #3          SET COUNT
047A 36       CERR   PSH A               SAVE COUNT
047B DF 77           STX     QTEMP3      SAVE PLACE
047D E6 00           LDA B   0,X         GET ERROR
047F 27 15           BEQ     CNXT        IF 0, GO NEXT
0481 96 56           LDA A   ERRFLG      GET FLAG
0483 27 0A           BEQ     PRT2ER
0485 96 AE           LDA A   LIST        CHECK LIST ON
0487 26 06           BNE     PRT2ER
0489 BD 05 FF        JSR     PRTDAT
048C BD 06 42        JSR     PRTSRC      PRINT INFO
048F DE 77    PRT2ER LDX     QTEMP3      GET POINTER
0491 E6 00           LDA B   0,X         GET ERROR
0493 BD 06 51        JSR     PRTERR      GO PRINT MESG
0496 DE 77    CNXT   LDX     QTEMP3      GET POINTER
0498 08              INX                 POINT NEXT
0499 32              PUL A   GET COUNT
049A 4A              DEC A               ONE DONE
049B 26 DD           BNE     CERR        LOOP TILL DONE
049D 96 5F    NOERR2 LDA A   PAGFLG      CHECK PAGE FLAG
049F 26 03           BNE     NOERR4
04A1 BD 11 31        JSR     EJECT
04A4 DE 6F    NOERR4 LDX     XTEMP3      GET SOURCE PTR
04A6 96 58           LDA A   ENDFLG
04A8 26 2C           BNE     FIN
04AA 9C 46           CPX     SRCEND      CHECK IF DONE
04AC 27 03           BEQ     P2DON
04AE 7E 03 E0        JMP     PASS2
04B1 39       P2DON  RTS
              *
              ** CONTRL
              * OUTPUT TAPE CONTROL CHARACTERS
04B2 C6 04    CONTRL LDA B   #4          SET 4 CHARS
04B4 27 09           BEQ     CONDON
04B6 A6 00    PCTRL  LDA A   0,X
04B8 BD 03 23        JSR     TAPOUT
04BB 08              INX
04BC 5A              DEC B
```

```
LOCN B1 B2 B3
04BD 26 F7              BNE     PCTRL
04BF 39         CONDON  RTS
04C0 00         TAPEON  FCB     0,0,0,0
04C1 00
04C2 00
04C3 00
04C4 00         TAPEOF  FCB     0,0,0,0
04C5 00
04C6 00
04C7 00
                *
                ** DELAY
                * DELAY FOR TAPE CONTROL
04C8 C6 04      DELAY   LDA B   #4
04CA 27 09              BEQ     DELDON
04CC CE F4 FF   XLOOP   LDX     #$F4FF      SET COUNTER
04CF 09         DECX    DEX
04D0 26 FD              BNE     DECX
04D2 5A                 DEC B
04D3 26 F7              BNE     XLOOP
04D5 39         DELDON  RTS
                *
                *
                *
                ** FIN
                * END OF ASSEMBLY CLEAN UP
04D6 96 5D      FIN     LDA A   P3FLG       CHECK PASS3
04D8 27 17              BEQ     LSTREC      IF SO, PUNCH LAST RECORD
04DA BD 07 BA           JSR     PCRLF       CR LF
04DD BD 06 39           JSR     PRT2
04E0 CE 05 49           LDX     #NOERHD
04E3 96 A9              LDA A   ERRORS      SEE IF ANY ERRORS
04E5 27 03              BEQ     PRTMES      IF NOT, GOT PTR
04E7 CE 05 4B           LDX     #ERRHD      MESSAGE
04EA BD 07 AB   PRTMES  JSR     PDATA       PRINT IT
04ED 96 B2      CHKTAP  LDA A   TAPE        SEE IF TAPE ON
04EF 27 14              BEQ     FIN2        IF NOT, SKIP
04F1 BD 15 18   LSTREC  JSR     PRTREC      GO PUNCH LAST
04F4 86 53              LDA A   #'S
04F6 BD 03 23           JSR     TAPOUT
04F9 86 39              LDA A   #'9
04FB BD 03 23           JSR     TAPOUT      PUNCH S9
04FE 8D C8              BSR     DELAY       DELAY BEFORE TURN OFF
0500 CE 04 C4           LDX     #TAPEOF
0503 8D AD              BSR     CONTRL
0505 96 5D      FIN2    LDA A   P3FLG       CHECK PASS3
0507 27 2E              BEQ     FIN6        IF SO, SKIP
0509 96 B3              LDA A   MEMORY      CHECK MEMORY OPTION
050B 27 09              BEQ     FIN5        IF OFF, SKIP
050D BD 15 F4           JSR     FIXCNT      GO SET BYTE COUNT
0510 DE 8B              LDX     MEMPTR      GET POINTER
0512 6F 00      SET0    CLR     0,X
0514 6F 01              CLR     1,X
0516 96 AF      FIN5    LDA A   SYMBOL      CHECK SYMBOL ON
0518 26 44              BNE     SYMGEN      IF SO, GO PRINT
```

```
LOCN B1 B2 B3
051A 96 AE              LDA A  LIST       SEE IF LIST ON
051C 27 19              BEQ    FIN6       IF NOT, SKIP
051E BD 07 BA  FIN3     JSR    PCRLF      CR LF
0521 96 B1              LDA A  PAGER      SEE IF PAGE ON
0523 27 0A              BEQ    FIN4       IF NOT, SKIP
0525 96 B1              LDA A  PAGER      SEE IF PAGE ON
0527 27 06              BEQ    FIN4       IF NOT, SKIP
0529 CE 11 D1           LDX    #EJSTR
052C 7E 07 AB           JMP    PDATA      PAGE EJECT
052F C6 04     FIN4     LDA B  #4
0531 BD 07 BA  GAPX     JSR    PCRLF
0534 5A                 DEC B
0535 26 FA              BNE    GAPX       PRINT 4 LINES
0537 39        FIN6     RTS               DONE
0538 20        SYMHD    FCC    '   SYMBOL TABLE:'
0539 20
053A 20
053B 53
053C 59
053D 4D
053E 42
053F 4F
0540 4C
0541 20
0542 54
0543 41
0544 42
0545 4C
0546 45
0547 3A
0548 04                 FCB    4
0549 4E        NOERHD   FCC    'NO'
054A 4F
054B 20        ERRHD    FCC    ' ERROR(S) DETECTED'
054C 45
054D 52
054E 52
054F 4F
0550 52
0551 28
0552 53
0553 29
0554 20
0555 44
0556 45
0557 54
0558 45
0559 43
055A 54
055B 45
055C 44
055D 04                 FCB    4
               *
               *
               ** SYMGEN
```

```
LOCN B1 B2 B3
                    * SORT AND PRINT SYMBOL TABLE
055E 96 5D          SYMGEN  LDA A   P3FLG       CHECK PASS 3
0560 27 BC                  BEQ     FIN3        IF SO, DONE
0562 C6 04                  LDA B   #4
0564 BD 0F D9               JSR     TYP11A      GO SPACE 4
0567 CE 05 38               LDX     #SYMHD
056A BD 07 AB               JSR     PDATA       PRINT HEADER
056D BD 13 F0               JSR     SHELL       GO SORT
0570 DE 40                  LDX     LBLBEG
0572 09                     DEX
0573 DF 69                  STX     XTEMP       SET POINTER
0575 BD 07 BA       LSTSYM  JSR     PCRLF
0578 C6 04                  LDA B   #4          SET 4 LABELS
057A DE 69          GETSYM  LDX     XTEMP       GET POINTER
057C 08                     INX
057D A6 00                  LDA A   0,X
057F 27 29                  BEQ     NOPRT       IF 0, NO LABEL
0581 37                     PSH B
0582 C6 06                  LDA B   #6          SET 6 CHARS
0584 A6 00          LABOUT  LDA A   0,X         GET CHAR
0586 BD 03 20               JSR     OUTCH       PRINT IT
0589 08                     INX
058A 5A                     DEC B               CHECK DONE
058B 26 F7                  BNE     LABOUT
058D BD 0C C7               JSR     OUT2S       PRINT 2 SPACES
0590 A6 00                  LDA A   0,X         GET MS ADDRESS
0592 BD 0C D0               JSR     OUTHEX      PRINT IT
0595 08                     INX
0596 A6 00                  LDA A   0,X         GET LS VALUE
0598 BD 0C D0               JSR     OUTHEX      PRINT IT
059B DF 69                  STX     XTEMP       SAVE PTR LOCATION
059D BD 06 39               JSR     PRT2        PRINT 7 SPACES
05A0 33                     PUL B               GET LINE COUNT
05A1 9C 42                  CPX     LBLEND      CHECK TABLE DONE
05A3 27 13                  BEQ     SYMPRT
05A5 5A             CONT    DEC B               SEE IF 4 YET
05A6 26 D2                  BNE     GETSYM      IF NOT, DO AGAIN
05A8 20 CB                  BRA     LSTSYM      OTHERWISE, START NEW LINE
05AA 37             NOPRT   PSH B
05AB C6 07                  LDA B   #7
05AD 08             MOVPTR  INX
05AE 5A                     DEC B
05AF 26 FC                  BNE     MOVPTR      ADVANCE PTR
05B1 33                     PUL B
05B2 DF 69                  STX     XTEMP       SAVE PTR
05B4 9C 42                  CPX     LBLEND      CHECK DONE
05B6 26 C2                  BNE     GETSYM
05B8 7E 05 1E       SYMPRT  JMP     FIN3
                    *
                    *
                    ** PASTHR
                    * PERFORM ASSEMBLY PASS 3
05BB 7F 00 5D       PASTHR  CLR     P3FLG       SET PASS 3
05BE 7E 03 D9               JMP     PASTWO      DO PASS 2
                    *
```

```
LOCN B1 B2 B3
                    ** PRTINF
                    * PRINT ASSEMBLED DATA
05C1 8D 3C      PRTINF  BSR     PRTDAT      GO PRINT ADDR, DATA
05C3 8D 7D              BSR     PRTSRC      PRINT SOURCE
05C5 CE 02 00           LDX     #BYTSTK
05C8 DF 71              STX     XTEMP4      SET MULTIPLE DATA PTR
05CA 96 5A              LDA A   DATFLG      CHECK MULTIPLE
05CC 26 01              BNE     PRTINA      IF SET, ITS THERE
05CE 39         PRTIND  RTS                 DONE
05CF 96 B0      PRTINA  LDA A   GENER       CHECK GENERATE FLAG
05D1 27 FB              BEQ     PRTIND      IF CLR, NO PRINT
05D3 96 90      PRTINE  LDA A   OPCNT       GET OPERAND COUNT
05D5 DE 6D      PRTINB  LDX     XTEMP2      GET OLD PC
05D7 08         PRTINC  INX                 BUMP
05D8 4A                 DEC A               DO UNTIL PAST PRINTED
05D9 26 FC              BNE     PRTINC
05DB DF 6D              STX     XTEMP2      SAVE NEW PRINTABLE PC
05DD 86 01              LDA A   #1
05DF 97 90              STA A   OPCNT       SET COUNT
05E1 DE 71              LDX     XTEMP4      GET STACK PTR
05E3 9C 87              CPX     BYTPTR      CHECK FOR DATA
05E5 27 E7              BEQ     PRTIND      IF NO DATA, EXIT
05E7 A6 00              LDA A   0,X         GET CHAR (BYTE)
05E9 97 7E              STA A   OPCODE      PUT IN PLACE
05EB 08                 INX                 BUMP POINTER
05EC 9C 87              CPX     BYTPTR      CHECK MORE DATA
05EE 27 08              BEQ     PRTING      IF NO, DONE
05F0 7C 00 90           INC     OPCNT       SET COUNT =2
05F3 A6 00              LDA A   0,X         GET NEXT BYTE
05F5 97 7F              STA A   OP1         PUT IN PLACE
05F7 08                 INX                 BUMP PTR
05F8 DF 71      PRTING  STX     XTEMP4      SAVE POINTER
05FA BD 05 FF           JSR     PRTDAT      GO PRINT DATA
05FD 20 D4              BRA     PRTINE      LOOP TILL DONE
                    *
                    *
                    ** PRTDAT
                    * PRINT ADDRESS AND DATA
05FF BD 07 BA   PRTDAT  JSR     PCRLF       GO DO CR LF
0602 BD 03 1E           JSR     OUTS        PRINT A SP
0605 96 59              LDA A   PCFLAG      CHECK FOR PRINT PC
0607 26 08              BNE     PRTPC       IF SET, DO IT
0609 BD 0C C7           JSR     OUT2S
060C BD 0C C5           JSR     OUT3S       SKIP FIELD
060F 20 25              BRA     PRT1
0611 96 6D      PRTPC   LDA A   XTEMP2      GET CURRENT PC
0613 BD 0C D0           JSR     OUTHEX      PRINT MS
0616 96 6E              LDA A   XTEMP2+1    GET LS
0618 BD 0C CC           JSR     OUTHXS      PRINT IT
061B D6 90              LDA B   OPCNT       GET COUNT
061D 27 17              BEQ     PRT1
061F 96 7E              LDA A   OPCODE
0621 BD 0C CC           JSR     OUTHXS      PRINT OPCODE
0624 5A                 DEC B
0625 27 12              BEQ     PRT2        SEE IF DONE
```

```
LOCN B1 B2 B3
0627 96 7F            LDA A  OP1
0629 BD 0C CC         JSR    OUTHXS     PRINT IT
062C 5A               DEC B
062D 27 0D            BEQ    PRT3
062F 96 80            LDA A  OP2
0631 BD 0C CC         JSR    OUTHXS
0634 20 09            BRA    PRT4
0636 BD 0C C5  PRT1   JSR    OUT3S
0639 BD 0C C5  PRT2   JSR    OUT3S
063C BD 0C C5  PRT3   JSR    OUT3S
063F 7E 03 1E  PRT4   JMP    OUTS
                      *
                      ** PRTSRC
                      * PRINT A LINE OF SOURCE
0642 DE 8D     PRTSRC LDX    LINPTR     GET POINTER
0644 A6 00     PRTS1  LDA A  0,X        GET A CHAR
0646 08               INX              POINT NEXT
0647 81 0D            CMP A  #$D       CHECK FOR CR
0649 27 05            BEQ    PRTS2     IF SO, DONE
064B BD 03 20         JSR    OUTCH     PRINT IT
064E 20 F4            BRA    PRTS1     DO AGAIN
0650 39        PRTS2  RTS              DONE
                      *
                      ** PRTERR
                      * INSERT ERROR MESSAGE INTO LISTING
0651 CE 06 81  PRTERR LDX    #MSGHD
0654 BD 07 B2         JSR    PSTR      PRINT HEADING
0657 7F 00 56         CLR    ERRFLG    SET PRINTED FLAG
065A CE 06 69         LDX    #MSGTBL   POINT TO TABLE
065D 58               ASL B            MULT ERROR # * 2
065E 27 04            BEQ    GOTMSG    CHECK IF GOT
0660 08        PTNXT  INX              POINT NEXT ADDRESS
0661 5A               DEC B            COUNT OFF
0662 26 FC            BNE    PTNXT     CYCLE
0664 EE 00     GOTMSG LDX    0,X       GET TEXT POINTER
0666 7E 07 AB         JMP    PDATA     GO PRINT MESG
                      *
0669 06 87     MSGTBL FDB    MESG0
066B 06 9D            FDB    MESG1
066D 06 AE            FDB    MESG2
066F 06 C6            FDB    MESG3
0671 06 DE            FDB    MESG4
0673 06 F9            FDB    MESG5
0675 07 16            FDB    MESG6
0677 07 2F            FDB    MESG7
0679 07 3C            FDB    MESG8
067B 07 53            FDB    MESG9
067D 07 78            FDB    MESG10
067F 07 8E            FDB    MESG11
                      *
0681 2A        MSGHD  FCC    '**
0682 2A
0683 20
0684 20
0685 20
```

```
LOCN B1 B2 B3
0686 04                 FCB     4
                *
0687 53         MESG0   FCC     'SYMBOL TABLE OVERFLOW'
0688 59
0689 4D
068A 42
068B 4F
068C 4C
068D 20
068E 54
068F 41
0690 42
0691 4C
0692 45
0693 20
0694 4F
0695 56
0696 45
0697 52
0698 46
0699 4C
069A 4F
069B 57
069C 04                 FCB     4
069D 55         MESG1   FCC     'UNDEFINED SYMBOL'
069E 4E
069F 44
06A0 45
06A1 46
06A2 49
06A3 4E
06A4 45
06A5 44
06A6 20
06A7 53
06A8 59
06A9 4D
06AA 42
06AB 4F
06AC 4C
06AD 04                 FCB     4
06AE 4D         MESG2   FCC     'MULTIPLY DEFINED SYMBOL'
06AF 55
06B0 4C
06B1 54
06B2 49
06B3 50
06B4 4C
06B5 59
06B6 20
06B7 44
06B8 45
06B9 46
06BA 49
06BB 4E
```

```
LOCN B1 B2 B3
06BC 45
06BD 44
06BE 20
06BF 53
06C0 59
06C1 4D
06C2 42
06C3 4F
06C4 4C
06C5 04                 FCB     4
06C6 55         MESG3   FCC     'UNRECOGNIZABLE MNEMONIC'
06C7 4E
06C8 52
06C9 45
06CA 43
06CB 4F
06CC 47
06CD 4E
06CE 49
06CF 5A
06D0 41
06D1 42
06D2 4C
06D3 45
06D4 20
06D5 4D
06D6 4E
06D7 45
06D8 4D
06D9 4F
06DA 4E
06DB 49
06DC 43
06DD 04                 FCB     4
06DE 49         MESG4   FCC     'ILLEGAL CHARACTER IN LABEL'
06DF 4C
06E0 4C
06E1 45
06E2 47
06E3 41
06E4 4C
06E5 20
06E6 43
06E7 48
06E8 41
06E9 52
06EA 41
06EB 43
06EC 54
06ED 45
06EE 52
06EF 20
06F0 49
06F1 4E
06F2 20
```

```
LOCN B1 B2 B3
06F3 4C
06F4 41
06F5 42
06F6 45
06F7 4C
06F8 04                 FCB     4
06F9 49         MESG5   FCC     'ILLEGAL CHARACTER IN OPERAND'
06FA 4C
06FB 4C
06FC 45
06FD 47
06FE 41
06FF 4C
0700 20
0701 43
0702 48
0703 41
0704 52
0705 41
0706 43
0707 54
0708 45
0709 52
070A 20
070B 49
070C 4E
070D 20
070E 4F
070F 50
0710 45
0711 52
0712 41
0713 4E
0714 44
0715 04                 FCB     4
0716 52         MESG6   FCC     'RELATIVE BRANCH TOO LONG'
0717 45
0718 4C
0719 41
071A 54
071B 49
071C 56
071D 45
071E 20
071F 42
0720 52
0721 41
0722 4E
0723 43
0724 48
0725 20
0726 54
0727 4F
0728 4F
0729 20
```

```
LOCN B1 B2 B3
072A 4C
072B 4F
072C 4E
072D 47
072E 04                 FCB     4
072F 53         MESG7   FCC     'SYNTAX ERROR'
0730 59
0731 4E
0732 54
0733 41
0734 58
0735 20
0736 45
0737 52
0738 52
0739 4F
073A 52
073B 04                 FCB     4
073C 49         MESG8   FCC     'ILLEGAL INDEX VARIABLE'
073D 4C
073E 4C
073F 45
0740 47
0741 41
0742 4C
0743 20
0744 49
0745 4E
0746 44
0747 45
0748 58
0749 20
074A 56
074B 41
074C 52
074D 49
074E 41
074F 42
0750 4C
0751 45
0752 04                 FCB     4
0753 49         MESG9   FCC     'ILLEGAL CHARACTER FOR SPECIFIED BASE'
0754 4C
0755 4C
0756 45
0757 47
0758 41
0759 4C
075A 20
075B 43
075C 48
075D 41
075E 52
075F 41
0760 43
```

```
LOCN B1 B2 B3
0761 54
0762 45
0763 52
0764 20
0765 46
0766 4F
0767 52
0768 20
0769 53
076A 50
076B 45
076C 43
076D 49
076E 46
076F 49
0770 45
0771 44
0772 20
0773 42
0774 41
0775 53
0776 45
0777 04                 FCB     4
0778 49         MESG10  FCC     'ILLEGAL OPTION SWITCH'
0779 4C
077A 4C
077B 45
077C 47
077D 41
077E 4C
077F 20
0780 4F
0781 50
0782 54
0783 49
0784 4F
0785 4E
0786 20
0787 53
0788 57
0789 49
078A 54
078B 43
078C 48
078D 04                 FCB     4
078E 54         MESG11  FCC     'TOO MANY OPERANDS (DATA)'
078F 4F
0790 4F
0791 20
0792 4D
0793 41
0794 4E
0795 59
0796 20
0797 4F
```

```
LOCN B1 B2 B3
0798 50
0799 45
079A 52
079B 41
079C 4E
079D 44
079E 53
079F 20
07A0 28
07A1 44
07A2 41
07A3 54
07A4 41
07A5 29
07A6 04                 FCB     4
                        *
                        ** PDATA
                        * PRINT STRINGS
07A7 BD 03 20   PLOOP   JSR     OUTCH       PRINT CHAR
07AA 08                 INX                 POINT NEXT
07AB A6 00      PDATA   LDA A   0,X         GET A CHAR
07AD 81 04              CMP A   #4          CHECK FOR EOT
07AF 26 F6              BNE     PLOOP       IF NOT,PRINT IT
07B1 39                 RTS                 DONE
                        *
                        ** PSTR
                        * PRINT CR,LF THEN STRING
07B2 DF 65      PSTR    STX     XSAVE       SAVE X
07B4 8D 04              BSR     PCRLF
07B6 DE 65              LDX     XSAVE       GET POINTER BAC K
07B8 20 F1              BRA     PDATA       GO PRINT IT
                        *
                        ** PCRLF
                        * PRINT CR AND LF
07BA CE 07 CF   PCRLF   LDX     #CRLF       POINT
07BD 8D EC              BSR     PDATA       GO PRINT
07BF 96 A8              LDA A   LINCNT      GET LINE COUNT
07C1 4C                 INC A
07C2 97 A8              STA A   LINCNT      BUMP IT
07C4 81 36              CMP A   #LINES      SEE IF TIME TO EJECT
07C6 22 04              BHI     PCRLF2      IF SO, GO DO IT
07C8 7F 00 5C   PCRLF1  CLR     EJFLG       CLEAR FLAG
07CB 39                 RTS                 DONE
07CC 7E 11 31   PCRLF2  JMP     EJECT       GO PAGE EJECT
07CF 0D         CRLF    FCB     $D,$A,0,0,0,0,4
07D0 0A
07D1 00
07D2 00
07D3 00
07D4 00
07D5 04
                        *
                        ** OPSERR
                        * FATAL ERROR ROUTINE
                        * GENERATES 3 NOP'S
```

```
LOCN B1 B2 B3
07D6 36          OPSERR  PSH A
07D7 86 01               LDA A  #01
07D9 97 7E               STA A  OPCODE
07DB 97 7F               STA A  OP1
07DD 97 80               STA A  OP2
07DF 97 59               STA A  PCFLAG    MAKE SURE PC ON
07E1 BD 0C 72            JSR    ADDPC3
07E4 32                  PUL A
                 *
                 ** ASMERR
                 * KEEP TRACK OF ASSEMBLY ERRORS
07E5 36          ASMERR  PSH A
07E6 97 84               STA A  LSTERR    SAVE ERROR
07E8 32                  PUL A
07E9 7D 00 56            TST    ERRFLG    CHECK ERROR SUPPRESS
07EC 26 33               BNE    ASME2     IF ON, DONT PROCESS
07EE C6 FF               LDA B  #$FF
07F0 D7 A9               STA B  ERRORS    SET FLAG
07F2 7D 00 8F            TST    PASS      CHECK PASS COUNT
07F5 26 2D               BNE    ASME3     IF NOT PASS1, SKIP
07F7 D6 A5               LDA B  ERRCNT    GET COUNT
07F9 C1 55               CMP B  #85       CHECK EXCESS
07FB 27 24               BEQ    ASME2     IF SO, IGNORE
07FD 36                  PSH A            SAVE ERROR #
07FE 96 4D               LDA A  SRCPTR    GET HIGH
0800 D6 4E               LDA B  SRCPTR+1  GET LOW
0802 DE 85               LDX    ERRPTR    GET STACK POINTER
0804 A7 00               STA A  0,X       STORE HIGH
0806 E7 01               STA B  1,X       STORE LOW
0808 32                  PUL A            GET ERROR #
0809 A7 02               STA A  2,X       SAVE #
080B 08                  INX
080C 08                  INX
080D 08                  INX              ADVANCE ERROR PTR
080E DF 85               STX    ERRPTR    SAVE IT
0810 96 A5               LDA A  ERRCNT    GET COUNT OF ERRORS
0812 4C                  INC A            KICK
0813 97 A5               STA A  ERRCNT
0815 81 55               CMP A  #85       ERROR LIMIT?
0817 26 08               BNE    ASME2
0819 CE 08 36            LDX    #TOOMAN
081C 8D 94               BSR    PSTR
081E 9E 67               LDS    SPSAVE    GET PROPER RET ADR.
0820 39                  RTS              DONE
0821 86 FF       ASME2   LDA A  #$FF
0823 39                  RTS              DONE
0824 D6 81       ASME3   LDA B  P2ERR1    CHECK EMPTY
0826 26 03               BNE    ASME4
0828 97 81               STA A  P2ERR1
082A 39                  RTS
082B D6 82       ASME4   LDA B  P2ERR2
082D 26 03               BNE    ASME5
082F 97 82               STA A  P2ERR2
0831 39                  RTS
0832 97 83       ASME5   STA A  P2ERR3
```

```
LOCN B1 B2 B3
0834 39                  RTS
0835 39                  RTS                    DONE
0836 45           TOOMAN FCC     'ERROR LIMIT EXCEEDED'
0837 52
0838 52
0839 4F
083A 52
083B 20
083C 4C
083D 49
083E 4D
083F 49
0840 54
0841 20
0842 45
0843 58
0844 43
0845 45
0846 45
0847 44
0848 45
0849 44
084A 04                  FCB    4
                  *
                  ** RANDOM
                  * RANDOM NUMBER GENERATOR USED FOR
                  * HASHING FUNCTION
084B 37           RANDOM PSH B                  SAVE B
084C 36                  PSH A                  AND A
084D C6 18               LDA B  #24             SET FOR 24 CYCLES
084F 96 91        LOOP   LDA A  RNDM           GET FIRST BYTE
0851 48                  ASL A
0852 48                  ASL A
0853 48                  ASL A
0854 98 91               EOR A  RNDM           XOR BIT 28 WITH 31
0856 48                  ASL A
0857 48                  ASL A                  GET RESULT IN CARRY
0858 79 00 93            ROL    RNDM+2
085B 79 00 92            ROL    RNDM+1
085E 79 00 91            ROL    RNDM           SHIFT ALL LEFT WITH C
0861 5A                  DEC B                 COUNT OFF
0862 26 EB               BNE    LOOP           LOOP UNTIL DONE
0864 32                  PUL A
0865 33                  PUL B
0866 39                  RTS
                  *
                  ** HASH
                  * HASH A SYMBOL TO A TABLE ADDRESS
0867 CE 00 4F     HASH   LDX    #LABEL         GET START OF LABEL
086A 7F 00 A4            CLR    HASHCT         SET HASH COUNTER TO 0
086D A6 00               LDA A  0,X            GET FIRST CHAR
086F AB 05               ADD A  5,X
0871 97 93               STA A  RNDM+2         FOLD THE LABEL
0873 A6 01               LDA A  1,X
0875 A9 04               ADC A  4,X
```

```
LOCN B1 B2 B3
0877 97 92              STA A  RNDM+1
0879 A6 02              LDA A  2,X
087B A9 03              ADC A  3,X
087D 97 91              STA A  RNDM       AND PUT IN RANDOM GEN
087F 7C 00 A4  REHASH   INC    HASHCT     KICK COUNTER
0882 BD 08 4B  MIX2     JSR    RANDOM     MIX EM UP
0885 96 93              LDA A  RNDM+2     GET RESULT
0887 84 F8              AND A  #$F8       FIX FOR 8 BYTES
0889 D6 92              LDA B  RNDM+1
088B C4 1F              AND B  #$1F       LIMIT TO 8K
088D 9B 41              ADD A  LBLBEG+1   ADD ON BEGINNING
088F D9 40              ADC B  LBLBEG     ADDRESS OF TABLE
0891 97 6A              STA A  XTEMP+1
0893 D7 69              STA B  XTEMP      SET EFFECTIVE ADDRESS
0895 D1 42              CMP B  LBLEND
0897 22 E9              BHI    MIX2
0899 25 04              BCS    MIX3
089B 91 43              CMP A  LBLEND+1
089D 22 E3              BHI    MIX2       SEE IF IN RANGE
089F DE 69     MIX3     LDX    XTEMP      GET THE ADDRESS
08A1 39                 RTS               DONE
                *
                ** PUTLBL
                * ENTER LABEL IN SYMBOL TABLE
08A2 8D C3      PUTLBL  BSR    HASH       GO HASH IT
08A4 A6 00      CHKFRE  LDA A  0,X        GET SYMBOL ENTRY
08A6 27 13              BEQ    PUTIT      IF FREE, TAKE IT
08A8 BD 08 DE           JSR    CHKLBL     GO SEE IF SAME
08AB 27 0B              BEQ    HERROR     IF SO, MULTIPLE OCCURENCE
08AD BD 08 7F           JSR    REHASH     GO REHASH ON COLLISION
08B0 96 A4              LDA A  HASHCT     GET COUNTER
08B2 81 28              CMP A  #40        IF 40 COLLISIONS, FULL
08B4 26 EE              BNE    CHKFRE     GO SEE IF FREE
08B6 86 00              LDA A  #0         SET ERROR 0
08B8 7E 07 E5  HERROR   JMP    ASMERR     GO REPORT ERROR
08BB 96 4F     PUTIT    LDA A  LABEL      GET CHAR
08BD A7 00              STA A  0,X        PUT IN TABLE
08BF 96 50              LDA A  LABEL+1
08C1 A7 01              STA A  1,X
08C3 96 51              LDA A  LABEL+2
08C5 A7 02              STA A  2,X
08C7 96 52              LDA A  LABEL+3
08C9 A7 03              STA A  3,X
08CB 96 53              LDA A  LABEL+4
08CD A7 04              STA A  4,X
08CF 96 54              LDA A  LABEL+5
08D1 A7 05              STA A  5,X
08D3 96 4B              LDA A  PC
08D5 A7 06              STA A  6,X        STORE PC (HI)
08D7 96 4C              LDA A  PC+1
08D9 A7 07              STA A  7,X        STORE PC (LO)
08DB DF 75              STX    LTEMP      SAVE LABEL ADDRESS
08DD 39                 RTS               DONE
                *
                ** CHKLBL
```

```
LOCN B1 B2 B3
                    * SEE IF LABELS MATCH
08DE 86 02          CHKLBL  LDA A  #2          SET ERROR
08E0 E6 00                  LDA B  0,X
08E2 D4 60                  AND B  LBLMSK
08E4 D1 4F                  CMP B  LABEL
08E6 26 1C                  BNE    CKDONE      IF NO, WERE OK
08E8 D6 50                  LDA B  LABEL+1
08EA E1 01                  CMP B  1,X
08EC 26 16                  BNE    CKDONE
08EE D6 51                  LDA B  LABEL+2
08F0 E1 02                  CMP B  2,X
08F2 26 10                  BNE    CKDONE
08F4 D6 52                  LDA B  LABEL+3
08F6 E1 03                  CMP B  3,X
08F8 26 0A                  BNE    CKDONE
08FA D6 53                  LDA B  LABEL+4
08FC E1 04                  CMP B  4,X
08FE 26 04                  BNE    CKDONE
0900 D6 54                  LDA B  LABEL+5
0902 E1 05                  CMP B  5,X
0904 39             CKDONE  RTS                DONE
                    *
                    ** FNDLBL
                    * FIND A LABEL IN SYMBOL TABLE
0905 BD 08 67       FNDLBL  JSR    HASH        GO HASH IT UP
0908 A6 00          FND10   LDA A  0,X         GET ENTRY
090A 27 0E                  BEQ    FERROR      IF EMPTY, NO FIND
090C BD 08 DE               JSR    CHKLBL      GO SEE IF MATCH
090F 27 0C                  BEQ    GOTLBL      IF SO, WE GOT IT
0911 BD 08 7F               JSR    REHASH      GO MIX EM UP AGAIN
0914 96 A4                  LDA A  HASHCT      GET COUNTER
0916 81 28                  CMP A  #40         IF DO 40 TIMES, NO GOOD
0918 26 EE                  BNE    FND10       RECYCLE
091A 86 FF          FERROR  LDA A  #$FF        SET ERROR
091C 39                     RTS
091D 4F             GOTLBL  CLR A  SET FLAG
091E 39                     RTS
                    *
                    ** FNDOPT
                    * FIND OPERATOR (TYPE) AND EXECUTE
091F 4F             FNDOPT  CLR A
0920 97 5A                  STA A  DATFLG
0922 97 57                  STA A  MATFLG
0924 97 5B                  STA A  FCCFLG
0926 97 5C                  STA A  EJFLG    CLEAR FLAGS
0928 DE 96                  LDX    OPNPTR   GET POINTER
092A DF 6B                  STX    XTEMP1   SET UP
092C DE 94                  LDX    OPTPTR     GET POINTER
092E A6 02                  LDA A  2,X        GET CHAR
0930 97 7D                  STA A  TEMP       SAVE 3RD CHAR
0932 E6 01                  LDA B  1,X        GET 2ND CHAR
0934 A6 00                  LDA A  0,X        GET 1ST CHAR
0936 CE 09 6B               LDX    #OPTABL    POINT TO TABLE
0939 A1 00          CHK1    CMP A  0,X        CHECK FOR MATCH
093B 27 15                  BEQ    MATCH1     IF SO, GO SEE NEXT
```

```
LOCN B1 B2 B3
093D 7D 00 57          TST     MATFLG      CHECK FLAG
0940 26 0B             BNE     OPTERR      IF SET, NO FIND
0942 08          NOMATL INX
0943 08                 INX
0944 08                 INX
0945 08                 INX
0946 08                 INX
0947 08                 INX
0948 8C 0B 75           CPX     #OPTEND+6   CHECK END TABLE
094B 26 EC              BNE     CHK1        IF NOT, CHECK NEXT
094D 86 03       OPTERR LDA A   #3          SET ERROR NO.
094F 7E 07 D6           JMP     OPSERR      GO REPORT
0952 97 57       MATCH1 STA A   MATFLG      SET FLAG
0954 E1 01              CMP B   1,X         CHECK 2ND MATCH
0956 26 EA              BNE     NOMATL      IF NOT, RESTART
0958 36                 PSH A               SAVE CHAR
0959 96 7D              LDA A   TEMP        GET 3RD
095B A1 02              CMP A   2,X         CHECK MATCH
095D 27 03              BEQ     BINGO       IF SO, GOT IT
095F 32                 PUL A               GET 1ST AGAIN
0960 20 E0              BRA     NOMATL
0962 32          BINGO  PUL A               FIX STACK
0963 A6 03              LDA A   3,X         GET OPCODE BASE
0965 97 7E              STA A   OPCODE      SAVE
0967 EE 04              LDX     4,X         GET TYPE ADDRESS
0969 6E 00              JMP     0,X         GO SERVICE TYPE
                 *
                 * THIS IS THE MNEMONIC RECOGNITION AND
                 * BASE OPCODE TABLE
096B 41          OPTABL FCC     'ABA'
096C 42
096D 41
096E 1B                 FCB     $1B
096F 0D 03              FDB     TYPE1
0971 41                 FCC     'ADC'
0972 44
0973 43
0974 89                 FCB     $89
0975 0D 51              FDB     TYPE5
0977 41                 FCC     'ADD'
0978 44
0979 44
097A 8B                 FCB     $8B
097B 0D 51              FDB     TYPE5
097D 41                 FCC     'AND'
097E 4E
097F 44
0980 84                 FCB     $84
0981 0D 51              FDB     TYPE5
0983 41                 FCC      'ASL'
0984 53
0985 4C
0986 48                 FCB     $48
0987 0D 7B              FDB     TYPE6
0989 41                 FCC     'ASR'
```

```
LOCN B1 B2 B3
098A 53
098B 52
098C 47                 FCB     $47
098D 0D 7B              FDB     TYPE6
098F 42                 FCC     'BCC'
0990 43
0991 43
0992 24                 FCB     $24
0993 0D 06              FDB     TYPE2
0995 42                 FCC     'BCS'
0996 43
0997 53
0998 25                 FCB     $25
0999 0D 06              FDB     TYPE2
099B 42                 FCC     'BEQ'
099C 45
099D 51
099E 27                 FCB     $27
099F 0D 06              FDB     TYPE2
09A1 42                 FCC     'BGE'
09A2 47
09A3 45
09A4 2C                 FCB     $2C
09A5 0D 06              FDB     TYPE2
09A7 42                 FCC     'BGT'
09A8 47
09A9 54
09AA 2E                 FCB     $2E
09AB 0D 06              FDB     TYPE2
09AD 42                 FCC     'BHI'
09AE 48
09AF 49
09B0 22                 FCB     $22
09B1 0D 06              FDB     TYPE2
09B3 42                 FCC     'BHS'
09B4 48
09B5 53
09B6 24                 FCB     $24
09B7 0D 06              FDB     TYPE2
09B9 42                 FCC     'BIT'
09BA 49
09BB 54
09BC 85                 FCB     $85
09BD 0D 51              FDB     TYPE5
09BF 42                 FCC     'BLE'
09C0 4C
09C1 45
09C2 2F                 FCB     $2F
09C3 0D 06              FDB     TYPE2
09C5 42                 FCC     'BLO'
09C6 4C
09C7 4F
09C8 25                 FCB     $25
09C9 0D 06              FDB     TYPE2
09CB 42                 FCC     'BLS'
```

```
LOCN B1 B2 B3
09CC 4C
09CD 53
09CE 23                  FCB    $23
09CF 0D 06               FDB    TYPE2
09D1 42                  FCC    'BLT'
09D2 4C
09D3 54
09D4 2D                  FCB    $2D
09D5 0D 06               FDB    TYPE2
09D7 42                  FCC    'BMI'
09D8 4D
09D9 49
09DA 2B                  FCB    $2B
09DB 0D 06               FDB    TYPE2
09DD 42                  FCC    'BNE'
09DE 4E
09DF 45
09E0 26                  FCB    $26
09E1 0D 06               FDB    TYPE2
09E3 42                  FCC    'BPL'
09E4 50
09E5 4C
09E6 2A                  FCB    $2A
09E7 0D 06               FDB    TYPE2
09E9 42                  FCC    'BRA'
09EA 52
09EB 41
09EC 20                  FCB    $20
09ED 0D 06               FDB    TYPE2
09EF 42                  FCC    'BSR'
09F0 53
09F1 52
09F2 8D                  FCB    $8D
09F3 0D 06               FDB    TYPE2
09F5 42                  FCC    'BVC'
09F6 56
09F7 43
09F8 28                  FCB    $28
09F9 0D 06               FDB    TYPE2
09FB 42                  FCC    'BVS'
09FC 56
09FD 53
09FE 29                  FCB    $29
09FF 0D 06               FDB    TYPE2
0A01 43                  FCC    'CBA'
0A02 42
0A03 41
0A04 11                  FCB    $11
0A05 0D 03               FDB    TYPE1
0A07 43                  FCC    'CLC'
0A08 4C
0A09 43
0A0A 0C                  FCB    $0C
0A0B 0D 03               FDB    TYPE1
0A0D 43                  FCC    'CLI'
```

```
LOCN B1 B2 B3
0A0E 4C
0A0F 49
0A10 0E                 FCB     $0E
0A11 0D 03              FDB     TYPE1
0A13 43                 FCC     'CLR'
0A14 4C
0A15 52
0A16 4F                 FCB     $4F
0A17 0D 7B              FDB     TYPE6
0A19 43                 FCC     'CLV'
0A1A 4C
0A1B 56
0A1C 0A                 FCB     $0A
0A1D 0D 03              FDB     TYPE1
0A1F 43                 FCC     'CMP'
0A20 4D
0A21 50
0A22 81                 FCB     $81
0A23 0D 51              FDB     TYPE5
0A25 43                 FCC     'COM'
0A26 4F
0A27 4D
0A28 43                 FCB     $43
0A29 0D 7B              FDB     TYPE6
0A2B 43                 FCC     'CPX'
0A2C 50
0A2D 58
0A2E 8C                 FCB     $8C
0A2F 0D 51              FDB     TYPE5
0A31 44                 FCC     'DAA'
0A32 41
0A33 41
0A34 19                 FCB     $19
0A35 0D 03              FDB     TYPE1
0A37 44                 FCC     'DEC'
0A38 45
0A39 43
0A3A 4A                 FCB     $4A
0A3B 0D 7B              FDB     TYPE6
0A3D 44                 FCC     'DES'
0A3E 45
0A3F 53
0A40 34                 FCB     $34
0A41 0D 03              FDB     TYPE1
0A43 44                 FCC     'DEX'
0A44 45
0A45 58
0A46 09                 FCB     $09
0A47 0D 03              FDB     TYPE1
0A49 45                 FCC     'END'
0A4A 4E
0A4B 44
0A4C 00                 FCB     00
0A4D 10 DD              FDB     TYPE16
0A4F 45                 FCC     'EOR'
```

```
LOCN B1 B2 B3
0A50 4F
0A51 52
0A52 88                     FCB     $88
0A53 0D 51                  FDB     TYPE5
0A55 45                     FCC     'EQU'
0A56 51
0A57 55
0A58 00                     FCB     0
0A59 10 B0                  FDB     TYPE15
0A5B 46                     FCC     'FCB'
0A5C 43
0A5D 42
0A5E 00                     FCB     0
0A5F 0F 42                  FDB     TYPE9
0A61 46                     FCC     'FCC'
0A62 43
0A63 43
0A64 00                     FCB     0
0A65 0E 87                  FDB     TYPE8
0A67 46                     FCC     'FDB'
0A68 44
0A69 42
0A6A 00                     FCB     0
0A6B 0F 7E                  FDB     TYPE10
0A6D 49                     FCC     'INC'
0A6E 4E
0A6F 43
0A70 4C                     FCB     $4C
0A71 0D 7B                  FDB     TYPE6
0A73 49                     FCC     'INS'
0A74 4E
0A75 53
0A76 31                     FCB     $31
0A77 0D 03                  FDB     TYPE1
0A79 49                     FCC     'INX'
0A7A 4E
0A7B 58
0A7C 08                     FCB     $08
0A7D 0D 03                  FDB     TYPE1
0A7F 4A                     FCC     'JMP'
0A80 4D
0A81 50
0A82 6E                     FCB     $6E
0A83 0D 35                  FDB     TYPE3
0A85 4A                     FCC     'JSR'
0A86 53
0A87 52
0A88 AD                     FCB     $AD
0A89 0D 35                  FDB     TYPE3
0A8B 4C                     FCC     'LDA'
0A8C 44
0A8D 41
0A8E 86                     FCB     $86
0A8F 0D 51                  FDB     TYPE5
0A91 4C                     FCC     'LDS'
```

```
LOCN B1 B2 B3
0A92 44
0A93 53
0A94 8E                 FCB     $8E
0A95 0D 51              FDB     TYPE5
0A97 4C                 FCC     'LDX'
0A98 44
0A99 58
0A9A CE                 FCB     $CE
0A9B 0D 51              FDB     TYPE5
0A9D 4C                 FCC     'LSR'
0A9E 53
0A9F 52
0AA0 44                 FCB     $44
0AA1 0D 7B              FDB     TYPE6
0AA3 4D                 FCC     'MON'
0AA4 4F
0AA5 4E
0AA6 00                 FCB     0
0AA7 10 DD              FDB     TYPE16
0AA9 4E                 FCC     'NAM'
0AAA 41
0AAB 4D
0AAC 00                 FCB     0
0AAD 10 E9              FDB     TYPE17
0AAF 4E                 FCC     'NEG'
0AB0 45
0AB1 47
0AB2 40                 FCB     $40
0AB3 0D 7B              FDB     TYPE6
0AB5 4E                 FCC     'NOP'
0AB6 4F
0AB7 50
0AB8 01                 FCB     01
0AB9 0D 03              FDB     TYPE1
0ABB 4F                 FCC     'OPT'
0ABC 50
0ABD 54
0ABE 00                 FCB     0
0ABF 0F ED              FDB     TYPE12
0AC1 4F                 FCC     'ORA'
0AC2 52
0AC3 41
0AC4 8A                 FCB     $8A
0AC5 0D 51              FDB     TYPE5
0AC7 4F                 FCC     'ORG'
0AC8 52
0AC9 47
0ACA 00                 FCB     0
0ACB 10 A2              FDB     TYPE14
0ACD 50                 FCC     'PAG'
0ACE 41
0ACF 47
0AD0 00                 FCB     0
0AD1 10 89              FDB     TYPE13
0AD3 50                 FCC     'PSH'
```

```
LOCN B1 B2 B3
0AD4 53
0AD5 48
0AD6 36                 FCB     $36
0AD7 0D 88              FDB     TYPE7
0AD9 50                 FCC     'PUL'
0ADA 55
0ADB 4C
0ADC 32                 FCB     $32
0ADD 0D 88              FDB     TYPE7
0ADF 52                 FCC     'RMB'
0AE0 4D
0AE1 42
0AE2 00                 FCB     0
0AE3 11 1F              FDB     TYPE18
0AE5 52                 FCC     'ROL'
0AE6 4F
0AE7 4C
0AE8 49                 FCB     $49
0AE9 0D 7B              FDB     TYPE6
0AEB 52                 FCC     'ROR'
0AEC 4F
0AED 52
0AEE 46                 FCB     $46
0AEF 0D 7B              FDB     TYPE6
0AF1 52                 FCC     'RTI'
0AF2 54
0AF3 49
0AF4 3B                 FCB     $3B
0AF5 0D 03              FDB     TYPE1
0AF7 52                 FCC     'RTS'
0AF8 54
0AF9 53
0AFA 39                 FCB     $39
0AFB 0D 03              FDB     TYPE1
0AFD 53                 FCC     'SBA'
0AFE 42
0AFF 41
0B00 10                 FCB     $10
0B01 0D 03              FDB     TYPE1
0B03 53                 FCC     'SBC'
0B04 42
0B05 43
0B06 82                 FCB     $82
0B07 0D 51              FDB     TYPE5
0B09 53                 FCC     'SEC'
0B0A 45
0B0B 43
0B0C 0D                 FCB     $0D
0B0D 0D 03              FDB     TYPE1
0B0F 53                 FCC     'SEI'
0B10 45
0B11 49
0B12 0F                 FCB     $0F
0B13 0D 03              FDB     TYPE1
0B15 53                 FCC     'SEV'
```

```
LOCN B1 B2 B3
0B16 45
0B17 56
0B18 0B                 FCB     $0B
0B19 0D 03              FDB     TYPE1
0B1B 53                 FCC     'SPC'
0B1C 50
0B1D 43
0B1E 00                 FCB     0
0B1F 0F BD              FDB     TYPE11
0B21 53                 FCC     'STA'
0B22 54
0B23 41
0B24 97                 FCB     $97
0B25 0D 54              FDB     TYPE4
0B27 53                 FCC     'STS'
0B28 54
0B29 53
0B2A 9F                 FCB     $9F
0B2B 0D 54              FDB     TYPE4
0B2D 53                 FCC     'STX'
0B2E 54
0B2F 58
0B30 DF                 FCB     $DF
0B31 0D 54              FDB     TYPE4
0B33 53                 FCC     'SUB'
0B34 55
0B35 42
0B36 80                 FCB     $80
0B37 0D 51              FDB     TYPE5
0B39 53                 FCC     'SWI'
0B3A 57
0B3B 49
0B3C 3F                 FCB     $3F
0B3D 0D 03              FDB     TYPE1
0B3F 54                 FCC     'TAB'
0B40 41
0B41 42
0B42 16                 FCB     $16
0B43 0D 03              FDB     TYPE1
0B45 54                 FCC     'TAP'
0B46 41
0B47 50
0B48 06                 FCB     $06
0B49 0D 03              FDB     TYPE1
0B4B 54                 FCC     'TBA'
0B4C 42
0B4D 41
0B4E 17                 FCB     $17
0B4F 0D 03              FDB     TYPE1
0B51 54                 FCC     'TPA'
0B52 50
0B53 41
0B54 07                 FCB     $07
0B55 0D 03              FDB     TYPE1
0B57 54                 FCC     'TST'
```

```
LOCN B1 B2 B3
0B58 53
0B59 54
0B5A 4D                 FCB     $4D
0B5B 0D 7B              FDB     TYPE6
0B5D 54                 FCC     'TSX'
0B5E 53
0B5F 58
0B60 30                 FCB     $30
0B61 0D 03              FDB     TYPE1
0B63 54                 FCC     'TTL'
0B64 54
0B65 4C
0B66 00                 FCB     0
0B67 10 E9              FDB     TYPE17
0B69 54                 FCC     'TXS'
0B6A 58
0B6B 53
0B6C 35                 FCB     $35
0B6D 0D 03              FDB     TYPE1
0B6F 57         OPTEND  FCC     'WAI'
0B70 41
0B71 49
0B72 3E                 FCB     $3E
0B73 0D 03              FDB     TYPE1
                ** PARSE
                * PARSE A LINE OF SOURCE INTO POINTERS
                * AND CHECK SYNTAX
0B75 96 48      PARSE   LDA A   LINBYT
0B77 08         PARS0A  INX
0B78 4A                 DEC A
0B79 2A FC              BPL     PARS0A
0B7B DF 7B              STX     QTEMP
0B7D DF 8D              STX     LINPTR      SAVE PRINT POSITION
0B7F 86 FF      PARSE0  LDA A   #$FF
0B81 97 55              STA A   PRFLG       SET PROCESS FLAG
0B83 97 5E              STA A   PRTFLG
0B85 97 5F              STA A   PAGFLG
0B87 BD 0C 65           JSR     CLRLAB      GO CLEAR LABEL STORE
0B8A 4F                 CLR A
0B8B 97 90              STA A   OPCNT       SET OP COUNT =0
0B8D 97 AB              STA A   MODFY       SET FLAG
0B8F 97 7D              STA A   TEMP
0B91 97 59              STA A   PCFLAG
0B93 97 81              STA A   P2ERR1
0B95 97 82              STA A   P2ERR2
0B97 97 83              STA A   P2ERR3
0B99 97 56              STA A   ERRFLG
0B9B DF 94              STX     OPTPTR
0B9D DF 96              STX     OPNPTR
0B9F DE 7B              LDX     QTEMP
0BA1 A6 00              LDA A   0,X         GET FIRST CHAR
0BA3 81 0D              CMP A   #$D         CHECK FOR EMPTY
0BA5 26 03              BNE     CHKCOM
0BA7 7E 0C 2D           JMP     PARSE3
0BAA 81 2A      CHKCOM  CMP A   #'*        CHECK FOR COMMEMT
```

```
LOCN B1 B2 B3
0BAC 27 78          BEQ    FINDCR
0BAE 81 20    PARSE1 CMP A  #'          CHECK FOR NO LABEL
0BB0 27 22          BEQ    PARSE2
0BB2 97 59          STA A  PCFLAG
0BB4 81 41          CMP A  #'A          CHECK FOR LETTER A
0BB6 25 04          BCS    LABERR
0BB8 81 5A          CMP A  #'Z          CHECK FOR Z
0BBA 23 07          BLS    PARS1A
0BBC 86 04    LABERR LDA A  #4          SET ERROR
0BBE BD 07 E5       JSR    ASMERR
0BC1 20 0E          BRA    PARS1B       FINISH LINE
0BC3 BD 0C 8F PARS1A JSR   COPLBL       GO COPY THE LABEL
0BC6 4D             TST A
0BC7 26 08          BNE    PARS1B
0BC9 C1 0D          CMP B  #$D          CHECK FOR CR
0BCB 27 60          BEQ    PARSE3
0BCD C1 20          CMP B  #'
0BCF 26 EB          BNE    LABERR
0BD1 BD 0C 50 PARS1B JSR   FINDS2       GO FIND A SPACE
0BD4 BD 0C 5C PARSE2 JSR   NXTBL2       GO GET NEXT TOKEN
0BD7 27 54          BEQ    PARSE3       IF Z, NO OPERATION
0BD9 5F             CLR B
0BDA D7 55          STA B  PRFLG        SET PROCESS FLAG
0BDC 86 FF          LDA A  #$FF
0BDE 97 59          STA A  PCFLAG
0BE0 DF 94          STX    OPTPTR       SAVE OPERATION POINTER
0BE2 08             INX
0BE3 A6 00          LDA A  0,X
0BE5 81 0D          CMP A  #$D
0BE7 27 16          BEQ    PARS2F
0BE9 08             INX
0BEA A6 00          LDA A  0,X
0BEC 81 0D          CMP A  #$D
0BEE 27 0F          BEQ    PARS2F
0BF0 20 12          BRA    PARS2A
0BF2 96 8F    PEVAL LDA A  PASS
0BF4 4A             DEC A
0BF5 97 56          STA A  ERRFLG
0BF7 BD 11 D5       JSR    EVAL         GO EVALUATE
0BFA 7F 00 56       CLR    ERRFLG
0BFD 39             RTS                 RETURN
0BFE 02             NOP                 SPACE
0BFF 86 03    PARS2F LDA A  #$03
0C01 20 48          BRA    PARFF2
0C03 02             NOP                 SPACE
0C04 8D 55    PARS2A BSR    NXTBLK
0C06 27 25          BEQ    PARSE3
0C08 81 41          CMP A  #'A          IS IT AN A?
0C0A 27 05          BEQ    PARS2D
0C0C 81 42          CMP A  #'B          IS IT A B?
0C0E 26 14          BNE    PARS2E
0C10 5C       PARS2B INC B
0C11 5C       PARS2D INC B
0C12 08             INX
0C13 A6 00          LDA A  0,X          GET CHAR
```

```
LOCN B1 B2 B3
0C15 81 0D              CMP A  #$D
0C17 27 20              BEQ    PARSE4
0C19 81 20              CMP A  #'          IS IT A SPACE?
0C1B 27 1F              BEQ    PARS2H
0C1D 09          PARS2J DEX
0C1E 20 04              BRA    PARS2E
0C20                    RMB    4
0C24 DF 96       PARS2E STX    OPNPTR
0C26 08          FINDCR INX               BUMP POINTER
0C27 A6 00              LDA A  0,X         GET CHAR
0C29 81 0D              CMP A  #$D         IS IT A CR
0C2B 26 F9              BNE    FINDCR      IF NOT, GET NEXT
0C2D 96 7D       PARSE3 LDA A  TEMP
0C2F 27 07              BEQ    PARSE5
0C31 DF 7B              STX    QTEMP
0C33 BD 07 D6    PARSE7 JSR    OPSERR
0C36 DE 7B       PARSE6 LDX    QTEMP
0C38 39          PARSE5 RTS
0C39 D7 AB       PARSE4 STA B  MODFY
0C3B 39                 RTS               DONE
0C3C D7 AB       PARS2H STA B  MODFY       SAVE
0C3E 8D 1C              BSR    NXTBL2      GET NEXT
0C40 27 EB              BEQ    PARSE3
0C42 20 E0              BRA    PARS2E
0C44 DE 4B       FND222 LDX    PC          GET PC
0C46 DF 6D              STX    XTEMP2      SAVE IT
0C48 7E 09 1F           JMP    FNDOPT
0C4B 97 7D       PARFF2 STA A  TEMP
0C4D 20 D7              BRA    FINDCR      GO LOCATE CR
0C4F 08          FINDSP INX               BUMP POINTER
0C50 A6 00       FINDS2 LDA A  0,X         GET THE CHAR
0C52 81 0D              CMP A  #$D         CHECK FOR CR
0C54 27 0E              BEQ    NXTBL3
0C56 81 20              CMP A  #'          IS IT A SPACE?
0C58 26 F5              BNE    FINDSP      IF NOT, GET NEXT
0C5A 39                 RTS               DONE
0C5B 08          NXTBLK INX               BUMP POINTER
0C5C A6 00       NXTBL2 LDA A  0,X         GET CHAR
0C5E 81 20              CMP A  #'          IS IT A SPACE?
0C60 27 F9              BEQ    NXTBLK      IF SO, GET NEXT
0C62 81 0D              CMP A  #$D         IS IT A CR
0C64 39          NXTBL3 RTS               DONE


                 *
                 ** CLRLAB
                 * CLEAR LABEL STORAGE
0C65 CE 00 20    CLRLAB LDX    #$0020
0C68 DF 4F              STX    LABEL
0C6A CE 20 20           LDX    #$2020
0C6D DF 51              STX    LABEL+2
0C6F DF 53              STX    LABEL+4     SET EM
0C71 39                 RTS
                 *
                 *
```

```
LOCN B1 B2 B3
                        *
                        ** ADDPCN
                        * INCREMENT PC N TIMES
                        * SET OPERAND (BYTE) COUNT
0C72 DE 4B      ADDPC3  LDX     PC          GET THE PC
0C74 08                 INX
0C75 08                 INX                 BUMP TWICE
0C76 7C 00 90           INC     OPCNT
0C79 7C 00 90           INC     OPCNT       KICK OPERAND COUNT
0C7C 20 0A              BRA     ADDPC0
0C7E DE 4B      ADDPC2  LDX     PC          GET THE PC
0C80 08                 INX                 BUMP IT
0C81 7C 00 90           INC     OPCNT
0C84 20 02              BRA     ADDPC0
0C86 DE 4B      ADDPC1  LDX     PC
0C88 08         ADDPC0  INX                 BUMP IT
0C89 DF 4B              STX     PC          PUT BACK
0C8B 7C 00 90           INC     OPCNT
0C8E 39                 RTS                 DONE
                        *
                        ** COPLBL
                        * COPY LABEL TO LABEL STORE
0C8F 8D 1B      COPLBL  BSR     GETCHR
0C91 97 4F              STA A   LABEL
0C93 8D 17              BSR     GETCHR
0C95 97 50              STA A   LABEL+1
0C97 8D 13              BSR     GETCHR
0C99 97 51              STA A   LABEL+2
0C9B 8D 0F              BSR     GETCHR
0C9D 97 52              STA A   LABEL+3
0C9F 8D 0B              BSR     GETCHR
0CA1 97 53              STA A   LABEL+4
0CA3 8D 07              BSR     GETCHR
0CA5 97 54              STA A   LABEL+5
0CA7 39                 RTS                 RETURN
0CA8 08         COPDON  INX
0CA9 39                 RTS
                        *
0CAA                    RMB     2
                        *
                        *
                        ** GETCHR
                        * GET A CHARACTER
0CAC A6 00      GETCHR  LDA A   0,X
0CAE 84 7F              AND A   #$7F        MASK  PARITY
0CB0 16                 TAB
0CB1 81 30              CMP A   #'0
0CB3 25 0C              BCS     FIX         IF <0, FIX STACK
0CB5 81 39              CMP A   #'9
0CB7 23 EF              BLS     COPDON      IF <=9, OK
0CB9 81 41              CMP A   #'A
0CBB 25 04              BCS     FIX         IF <A, FIX STACK
0CBD 81 5A              CMP A   #'Z
0CBF 23 E7              BLS     COPDON      IF <=Z, OK
0CC1 31         FIX     INS
```

```
LOCN B1 B2 B3
0CC2 31          INS             FIX STACK
0CC3 4F          CLR A           SET A
0CC4 39          RTS             DONE
                 *
                 ** OUT3S
                 * PRINT 3 SPACES
0CC5 8D 02       OUT3S   BSR     OUTSZ
0CC7 8D 00       OUT2S   BSR     OUTSZ
0CC9 7E 03 1E    OUTSZ   JMP     OUTS    PRINT A SPACE
                 *
                 ** OUTHXS
                 * PRINT 2 HEX DIGITS AND A SPACE
0CCC 8D 02       OUTHXS  BSR     OUTHEX  GO PRINT DIGITS
0CCE 20 F9               BRA     OUTSZ
                 *
                 ** OUTHEX
                 * PRINT A AS 2 HEX DIGITS
0CD0 36          OUTHEX  PSH A           SAVE
0CD1 8D 08               BSR     HEXL    GO CONVERT
0CD3 8D 03               BSR     PRTIT   GO PRINT IT
0CD5 32                  PUL A
0CD6 8D 07               BSR     HEXR    GO CONVERT
0CD8 7E 03 20    PRTIT   JMP     OUTCH   GO PRINT
                 *
0CDB 44          HEXL    LSR A
0CDC 44                  LSR A
0CDD 44                  LSR A
0CDE 44                  LSR A
                 *
0CDF 84 0F       HEXR    AND A   #$F     MASK DOWN
0CE1 8B 90               ADD A   #$90
0CE3 19                  DAA
0CE4 89 40               ADC A   #$40
0CE6 19                  DAA
0CE7 39                  RTS             TRICK CONVERT
                 *
                 ** SUB16
                 * 16 BIT SUBTRACT
0CE8 97 7D       SUB16   STA A   TEMP    SAVE
0CEA A6 01               LDA A   1,X
0CEC 10                  SBA
0CED A7 01               STA A   1,X
0CEF A6 00               LDA A   0,X
0CF1 92 7D               SBC A   TEMP
0CF3 A7 00               STA A   0,X
0CF5 49                  ROL A
0CF6 88 01               EOR A   #1
0CF8 46                  ROR A           SET ARITH CARRY
0CF9 39                  RTS
                 *
                 ** ADD16
                 * 16 BIT ADD
0CFA EB 01       ADD16   ADD B   1,X     ADD ON
0CFC A9 00               ADC A   0,X     ADD WITH CARRY (MS)
0CFE A7 00               STA A   0,X     SAVE
```

```
LOCN B1 B2 B3
0D00 E7 01              STA B  1,X        SAVE LS
0D02 39                 RTS
                  *
                  ** TYPE1
                  * HANDLES TYPE1 INSTRUCTIONS
0D03 7E 0C 86     TYPE1 JMP    ADDPC1     GO FIX PC
                  *
                  ** TYPE2
                  * HANDLES TYPE2 INSTRUCTIONS
0D06 96 AB        TYPE2 LDA A  MODFY      CHECK MODIFY FLAG
0D08 26 42              BNE    TYP3R
0D0A BD 0C 7E           JSR    ADDPC2
0D0D 96 8F              LDA A  PASS       CHECK PASS COUNT
0D0F 27 23              BEQ    TYPE2D     IF PASS 1, SKIP
0D11 BD 11 D5           JSR    EVAL       GO EVALUATE OPERAND
0D14 26 16              BNE    TYPE2B     IF EVAL ERROR, UNDEFINED
0D16 96 4B              LDA A  PC
0D18 D6 4C              LDA B  PC+1       REFERENCE ADDRESS
0D1A CE 00 7B           LDX    #QTEMP     POINT
0D1D BD 0C E8           JSR    SUB16      GO SUBTRACT
0D20 4F                 CLR A
0D21 D6 7C              LDA B  QTEMP+1    GET LS RESULT
0D23 D7 7F              STA B  OP1        SAVE BRANCH AMOUNT
0D25 2A 01              BPL    TYPE2A     IF POS, SKIP
0D27 43                 COM A             COMPLEMENT A
0D28 91 7B        TYPE2A CMP A  QTEMP      CHECK SIGN EXTENSION
0D2A 27 08              BEQ    TYPE2D     IF EQUAL, OK
0D2C 7F 00 7F     TYPE2B CLR    OP1        SET BRANCH = 0
0D2F 86 06              LDA A  #6
0D31 7E 07 E5           JMP    ASMERR     GO REPORT ERROR
0D34 39           TYPE2D RTS               DONE
                  *
                  ** TYPE3
                  * HANDLES TYPE3 INST.
0D35 96 AB        TYPE3 LDA A  MODFY      GET MODIFIER
0D37 26 13              BNE    TYP3R      IF SET, ERROR
0D39 BD 0D A3     TYPE3A JSR   INDEX      GO CHECK INDEXED
                  *
                  ** EXTEND
                  * CHECKS FOR EXTENDED ADDRESSING (DEFAULT)
0D3C 96 8F        EXTEND LDA A  PASS
0D3E 27 09              BEQ    EXTEN1     CHECK PASS=1
0D40 BD 11 D5           JSR    EVAL       GO EVALUATE OPERAND
0D43 DE 7B        EXTEN0 LDX    QTEMP      GET RESULT
0D45 DF 7F              STX    OP1        SET BYTES 2,3
0D47 8D 13              BSR    FIXMOD
0D49 7E 0C 72     EXTEN1 JMP    ADDPC3     KICK PC AGAIN
0D4C 86 03        TYP3R  LDA A  #3
0D4E 7E 07 D6           JMP    OPSERR
                  *
                  ** TYPE5
                  * HANDLE TYPE5 INST.
0D51 BD 0E 3F     TYPE5  JSR    IMMED      CHECK IMMEDIATE
                  *
                  ** TYPE4
```

```
LOCN B1 B2 B3
                    * HANDLE TYPE4 INST.
0D54 BD 0E 04       TYPE4   JSR     DIRECT      GO CHECK DIRECT
0D57 20 E0                  BRA     TYPE3A      DEFAULT EXTEND
                    *
                    ** FIXMOD
                    * SET UP MODIFIER
0D59 8D 01          TFIXMD  BSR     FIXMOD
0D5B 39                     RTS
0D5C D6 7E          FIXMOD  LDA B   OPCODE
0D5E C1 80                  CMP B   #$80
0D60 24 05                  BCC     FIXM3       CHECK NO MODIFIER
0D62 96 AB          FIXM4   LDA A   MODFY
0D64 26 36                  BNE     TYPE7C      CHECK ILLEGAL
0D66 39                     RTS
0D67 C4 0F          FIXM3   AND B   #$F
0D69 C1 0B                  CMP B   #$B         CHECK NO MODIFIER
0D6B 22 F5                  BHI     FIXM4
0D6D 96 AB          FIXM5   LDA A   MODFY       GET MODIFIER
0D6F 27 2B                  BEQ     TYPE7C
0D71 4A                     DEC A
0D72 40                     NEG A
0D73 84 40                  AND A   #$40
0D75 9B 7E                  ADD A   OPCODE
0D77 97 7E                  STA A   OPCODE      FIX UP OPCODE
0D79 4F                     CLR A               RESET ERROR
0D7A 39                     RTS
                    *
                    ** TYPE6
                    * HANDLE TYPE6 INST.
0D7B 96 AB          TYPE6   LDA A   MODFY       GET MODIFIER
0D7D 4A                     DEC A
0D7E 2A 0D                  BPL     TYPE7A      CHECK INHERENT (A,B)
0D80 D6 7E                  LDA B   OPCODE      GET OPCODE
0D82 CB 20                  ADD B   #$20        ADD ON
0D84 D7 7E                  STA B   OPCODE      PUT BACK
0D86 20 B1                  BRA     TYPE3A      GO DO TYPE3
                    *
                    ** TYPE7
                    * HANDLE TYPE7 INSTRUCTIONS
0D88 96 AB          TYPE7   LDA A   MODFY       GET MODIFIER
0D8A 4A                     DEC A
0D8B 2B BF,                 BMI     TYP3R
0D8D D6 7E          TYPE7A  LDA B   OPCODE      GET CODE
0D8F C1 3F                  CMP B   #$3F        CHECK PUSH OR PULL
0D91 23 03                  BLS     TYPE7D
0D93 40                     NEG A
0D94 84 10                  AND A   #$10        MASK DOWN
0D96 1B             TYPE7D  ABA                 MODIFY
0D97 97 7E                  STA A   OPCODE      SAVE
0D99 7E 0C 86               JMP     ADDPC1      KICK PC
0D9C 31             TYPE7C  INS
0D9D 31                     INS
0D9E 86 03                  LDA A   #3
0DA0 7E 07 D6               JMP     OPSERR
                    *
```

```
LOCN B1 B2 B3
                    ** INDEX
                    * CHECK FOR INDEX ADDRESSING
                    * RETURN IF NOT
0DA3 DE 6B          INDEX   LDX     XTEMP1      GET OPERAND PTR
0DA5 7F 00 7F               CLR     OP1
0DA8 A6 00                  LDA A   0,X         FIRST CHAR
0DAA 81 58                  CMP A   #'X         IS IT AN X?
0DAC 26 0C                  BNE     INDEX1      IF NOT, CHECK NEXT
0DAE A6 01                  LDA A   1,X
0DB0 81 20                  CMP A   #'
0DB2 27 22                  BEQ     INDEX3
0DB4 81 0D                  CMP A   #$D
0DB6 26 02                  BNE     INDEX1
0DB8 20 1C                  BRA     INDEX3
0DBA A6 00          INDEX1  LDA A   0,X         GET CHAR
0DBC 81 2C                  CMP A   #',         CHECK FOR COMMA
0DBE 27 20                  BEQ     INDEX4
0DC0 81 20                  CMP A   #'          CHECK FOR SPACE
0DC2 27 2F                  BEQ     INDEX0      IF SO, EXTENDED
0DC4 81 0D                  CMP A   #$D
0DC6 27 2B                  BEQ     INDEX0
0DC8 08                     INX
0DC9 20 EF                  BRA     INDEX1
0DCB 96 8F          INDEX2  LDA A   PASS
0DCD 27 07                  BEQ     INDEX3      CHECK PASS COUNT
0DCF BD 11 D5               JSR     EVAL        GO EVALUATE
0DD2 96 7C                  LDA A   QTEMP+1
0DD4 97 7F                  STA A   OP1         SET OFFSET
0DD6 BD 0D 59       INDEX3  JSR     TFIXMD
0DD9 26 26                  BNE     FIXXX2
0DDB 31                     INS
0DDC 31                     INS                 FIX STAC
0DDD 7E 0C 7E               JMP     ADDPC2
0DE0 A6 01          INDEX4  LDA A   1,X         GET NEXT CHAR
0DE2 81 58                  CMP A   #'X         IS IT X
0DE4 26 14                  BNE     INDEX5      IF NOT, EXTENDED
0DE6 08                     INX
0DE7 A6 01          INDE00  LDA A   1,X         GET FOLLOWING
0DE9 81 20                  CMP A   #'          MUST BE SPACE
0DEB 27 DE                  BEQ     INDEX2      IF SO, INDEXED
0DED 81 0D                  CMP A   #$D
0DEF 27 DA                  BEQ     INDEX2
0DF1 20 07                  BRA     INDEX5
0DF3 D6 7E          INDEX0  LDA B   OPCODE
0DF5 CB 10                  ADD B   #$10
0DF7 D7 7E                  STA B   OPCODE
0DF9 39             INDEX9  RTS
0DFA 86 08          INDEX5  LDA A   #8
0DFC 31                     INS
0DFD 31                     INS
0DFE 7E 07 D6               JMP     OPSERR      GO REPORT ERROR
0E01 31             FIXXX2  INS
0E02 31                     INS                 FIX STACK
0E03 39                     RTS                 DONE
                    *
```

```
LOCN B1 B2 B3
                    ** DIRECT
                    * CHECK FOR DIRECT ADDRESSING
0E04 DE 6B          DIRECT  LDX     XTEMP1
0E06 86 FF                  LDA A   #$FF
0E08 97 56                  STA A   ERRFLG      DISABLE ERRORS
0E0A 97 60                  STA A   LBLMSK      SET MASK
0E0C DF 73                  STX     XTEMP5      SAVE POINTER
0E0E BD 11 D5               JSR     EVAL        GO CALCULATE
0E11 7F 00 56               CLR     ERRFLG      ENABLE ERRORS
0E14 C6 7F                  LDA B   #$7F
0E16 D7 60                  STA B   LBLMSK      RESET MASK
0E18 DE 6B                  LDX     XTEMP1      GET END PTR
0E1A E6 00                  LDA B   0,X         GET TERMINATOR
0E1C C1 2C                  CMP B   #',         CHECK  INDEXED
0E1E 36                     PSH A
0E1F 07                     TPA
0E20 DE 73                  LDX     XTEMP5
0E22 DF 6B                  STX     XTEMP1
0E24 33                     PUL B
0E25 06                     TAP                 RESET CCR
0E26 27 10                  BEQ     NDIR
0E28 5D                     TST B
0E29 26 0D                  BNE     NDIR        IF NO FIND ALL, NO DIRECT
0E2B D6 7B                  LDA B   QTEMP       GET MS BYTE
0E2D 26 09                  BNE     NDIR
0E2F BD 0D 59               JSR     TFIXMD
0E32 26 50                  BNE     FIXXX
0E34 96 7C                  LDA A   QTEMP+1
0E36 20 2F                  BRA     IMMED2
0E38 D6 7E          NDIR    LDA B   OPCODE
0E3A CB 10                  ADD B   #$10
0E3C D7 7E                  STA B   OPCODE
0E3E 39                     RTS                 PASS ON
                    *
                    ** IMMED
                    * CHECK FOR IMMEDIATE ADDRESSING
0E3F DE 6B          IMMED   LDX     XTEMP1      GET OPERAND PTR
0E41 A6 00                  LDA A   0,X
0E43 81 23                  CMP A   #'#         CHECK FOR #
0E45 27 07                  BEQ     IMMED1      IF SO, IMMEDIATE
0E47 D6 7E          IMMED0  LDA B   OPCODE
0E49 CB 10                  ADD B   #$10
0E4B D7 7E                  STA B   OPCODE
0E4D 39                     RTS
0E4E 08             IMMED1  INX
0E4F DF 6B                  STX     XTEMP1      MOVE PAST #
0E51 D6 7E                  LDA B   OPCODE
0E53 C4 0F                  AND B   #$F
0E55 C1 0B                  CMP B   #$B
0E57 22 15                  BHI     IMMED3
0E59 BD 0D 59               JSR     TFIXMD
0E5C 26 26                  BNE     FIXXX
0E5E 96 8F                  LDA A   PASS
0E60 27 07                  BEQ     IMMED4      IF PASS1, SKIP
0E62 BD 11 D5               JSR     EVAL        GO EVALUATE OPERAND
```

```
LOCN B1 B2 B3
0E65 96 7C            LDA A  QTEMP+1   GET LS RESULT
0E67 97 7F      IMMED2 STA A  OP1       SET BYTE 2
0E69 BD 0C 7E   IMMED4 JSR    ADDPC2
0E6C 20 16             BRA    FIXXX
0E6E 96 AB      IMMED3 LDA A  MODFY
0E70 4A                DEC A
0E71 2B 03             BMI    IMMED5
0E73 7E 0D 9C   IMMED6 JMP    TYPE7C
0E76 BD 0C 72   IMMED5 JSR    ADDPC3
0E79 96 8F             LDA A  PASS
0E7B 27 07             BEQ    FIXXX     CHECK PASS COUNT
0E7D BD 11 D5          JSR    EVAL      GO EVALUATE
0E80 DE 7B             LDX    QTEMP     GET ARG
0E82 DF 7F             STX    OP1       SET OPERANDS
0E84 31         FIXXX  INS
0E85 31                INS
0E86 39                RTS
                *
                ** TYPE8
                *
0E87 86 FF      TYPE8  LDA A  #$FF
0E89 97 56             STA A  ERRFLG    SUPPRESS ERROR REPORT
0E8B DE 96             LDX    OPNPTR
0E8D DF 73             STX    XTEMP5    SAVE START
0E8F BD 11 D5          JSR    EVAL      GO EVALUATE EXPR
0E92 CE 02 00          LDX    #BYTSTK
0E95 DF 87             STX    BYTPTR    SET UP POINTER
0E97 96 7C             LDA A  QTEMP+1   GET RESULT
0E99 27 56             BEQ    TYPE8F    IF ZERO, DELIM TYPE
0E9B DE 6B             LDX    XTEMP1
0E9D A6 00             LDA A  0,X
0E9F 81 2C             CMP A  #',
0EA1 26 4E             BNE    TYPE8F    IF NOT COMMA, DELIM TYPE
0EA3 08                INX               MOVE PAST ,
0EA4 96 7C             LDA A  QTEMP+1   GET DATA
0EA6 E6 00             LDA B  0,X       GET NEXT CHAR
0EA8 08                INX
0EA9 C1 0D             CMP B  #$D        CHECK FOR CR
0EAB 26 04             BNE    TYPE8A
0EAD 97 5B             STA A  FCCFLG
0EAF C6 20             LDA B  #$20       GET SPACE
0EB1 D7 7E      TYPE8A STA B  OPCODE     STORE FIRST BYTE
0EB3 DF 71             STX    XTEMP4    SAVE PTR
0EB5 BD 0C 86          JSR    ADDPC1    KICK PC
0EB8 DE 71             LDX    XTEMP4    GET PTR BACK
0EBA 4A                DEC A            SEE IF DONE
0EBB 26 01             BNE    TYPE8B
0EBD 39                RTS
0EBE 97 5A      TYPE8B STA A  DATFLG     SET FLAG
0EC0 86 01             LDA A  #1
0EC2 97 A6             STA A  BYTCNT     SAVE BYTE COUNT
0EC4 E6 00      TYPE8E LDA B  0,X        GET CHAR
0EC6 08                INX               KICK PTR
0EC7 DF 71             STX    XTEMP4    SAVE
0EC9 7D 00 5B          TST    FCCFLG    CHECK FLAG
```

```
LOCN B1 B2 B3
0ECC 26 06              BNE     TYPE8D
0ECE C1 0D              CMP B   #$D          CHECK CR
0ED0 26 04              BNE     TYPE8C
0ED2 97 5B              STA A   FCCFLG
0ED4 C6 20      TYPE8D  LDA B   #$20
0ED6 DE 87      TYPE8C  LDX     BYTPTR       GET STACK PTR
0ED8 E7 00              STA B   0,X          PUT ON STACK
0EDA 08                 INX
0EDB DF 87              STX     BYTPTR       SAVE NEXT POSITION
0EDD BD 0C 86           JSR     ADDPC1
0EE0 DE 71              LDX     XTEMP4       RETRIEVE PTR
0EE2 7A 00 5A           DEC     DATFLG       COUNT OFF
0EE5 26 DD              BNE     TYPE8E       LOOP TILL DONE
0EE7 86 01              LDA A   #1
0EE9 97 90              STA A   OPCNT    CORRECT OP COUNT
0EEB 97 5A              STA A   DATFLG     SET FLAG
0EED 7F 00 56           CLR     ERRFLG     CLEAR ERROR SUPPRESS
0EF0 39                 RTS                DONE
                *
0EF1 DE 73      TYPE8F  LDX     XTEMP5   GET START POINTER
0EF3 E6 00              LDA B   0,X        GET DELIMITER
0EF5 08                 INX                MOVE PAST
0EF6 A6 00              LDA A   0,X        GET CHAR
0EF8 97 7E              STA A   OPCODE     PUT AWAY
0EFA DF 71              STX     XTEMP4
0EFC BD 0C 86           JSR     ADDPC1     KICK PC
0EFF DE 71              LDX     XTEMP4
0F01 E1 01              CMP B   1,X        CHECK END
0F03 26 01              BNE     TYPE8G
0F05 39                 RTS
0F06 D7 5A      TYPE8G  STA B   DATFLG      SET FLAG
0F08 86 01              LDA A   #1
0F0A 97 A6              STA A   BYTCNT     SET COUNT
0F0C 08                 INX                MOVE POINTER
0F0D A6 00      TYPE8H  LDA A   0,X         GET CHAR
0F0F 08                 INX
0F10 DF 71              STX     XTEMP4     SAVE PTR
0F12 DE 87              LDX     BYTPTR     GET STACK PTR
0F14 11                 CBA                CHECK END
0F15 27 15              BEQ     TYPE8I     IF SO, QUIT
0F17 81 0D              CMP A   #$D        CHECK FOR CR
0F19 27 11              BEQ     TYPE8I     IF SO, QUIT
0F1B A7 00              STA A   0,X        PUT ON STACK
0F1D 08                 INX
0F1E DF 87              STX     BYTPTR     SAVE NEW POSITION
0F20 8C 03 00           CPX     #BYTSTK+256
0F23 27 13              BEQ     TYPE8J
0F25 BD 0C 86           JSR     ADDPC1
0F28 DE 71              LDX     XTEMP4     GET SOURCE PTR BACK
0F2A 20 E1              BRA     TYPE8H     LOOP TILL DONE
0F2C 7F 00 56   TYPE8I  CLR     ERRFLG     RESET ERROR SUPPRESSION
0F2F 86 01              LDA A   #1
0F31 97 90              STA A   OPCNT      SET COUNT
0F33 39                 RTS                DONE
                *
```

```
LOCN B1 B2 B3
0F34 8D 63      TYPE8K  BSR     TYP10C
0F36 20 02              BRA     TYPE8L
0F38 8D F2      TYPE8J  BSR     TYPE8I
0F3A 7F 00 56   TYPE8L  CLR     ERRFLG      RESET FLAG
0F3D 86 0B              LDA A   #11         SET ERROR
0F3F 7E 07 E5           JMP     ASMERR
                *
                *
                ** TYPE9
                * HANDLES TYPE 9 INSTRUCTIONS
0F42 CE 02 00   TYPE9   LDX     #BYTSTK
0F45 DF 87              STX     BYTPTR      SET UP STACK
0F47 BD 0B F2           JSR     PEVAL       GO EVALUATE
0F4A 96 7C              LDA A   QTEMP+1     GET DATA
0F4C 97 7E              STA A   OPCODE      PUT AWAY
0F4E BD 0C 86   TYPE9C  JSR     ADDPC1      KICK PC
0F51 DE 6B              LDX     XTEMP1      GET SOURCE PTR
0F53 A6 00              LDA A   0,X
0F55 81 0D              CMP A   #$D         CHECK DONE
0F57 27 04              BEQ     TYPE9D
0F59 81 2C              CMP A   #',
0F5B 27 05              BEQ     TYPE9A
0F5D 86 01      TYPE9D  LDA A   #1
0F5F 97 90              STA A   OPCNT    CORRECT COUNT
0F61 39                 RTS
0F62 97 5A      TYPE9A  STA A   DATFLG      SET
0F64 86 01              LDA A   #1
0F66 97 A6              STA A   BYTCNT      SET COUNT
0F68 08         TYPE9B  INX                 MOVE TO NEXT ARGUMENT
0F69 DF 6B              STX     XTEMP1      SAVE PTR
0F6B BD 0B F2           JSR     PEVAL       GO CRUNCH
0F6E DE 87              LDX     BYTPTR      GET STACK
0F70 96 7C              LDA A   QTEMP+1     GET DATA
0F72 A7 00              STA A   0,X         SAVE IT
0F74 08                 INX
0F75 DF 87              STX     BYTPTR      UPDATE AND SAVE
0F77 8C 03 00           CPX     #BYTSTK+256
0F7A 27 BC              BEQ     TYPE8J
0F7C 20 D0              BRA     TYPE9C
                *
                *
                ** TYPE10
                * EVALUATE TYPE 10 INSTRUCTION
0F7E CE 02 00   TYPE10  LDX     #BYTSTK
0F81 DF 87              STX     BYTPTR      SET UP STACK
0F83 BD 0B F2           JSR     PEVAL       GO EVALUATE
0F86 DE 7B              LDX     QTEMP
0F88 DF 7E              STX     OPCODE      PUT DATA
0F8A BD 0C 7E   TYP10A  JSR     ADDPC2      KICK PC
0F8D DE 6B              LDX     XTEMP1      GET TERM PTR
0F8F A6 00              LDA A   0,X         GET TERM
0F91 81 0D              CMP A   #$D         CHECK CR
0F93 27 04              BEQ     TYP10C
0F95 81 2C              CMP A   #',
0F97 27 05              BEQ     TYP10B
```

```
LOCN B1 B2 B3
0F99 86 02      TYP10C  LDA A  #2
0F9B 97 90              STA A  OPCNT    CORRECT COUNT
0F9D 39                 RTS
0F9E 97 5A      TYP10B  STA A  DATFLG    SET MULT DATA FLAG
0FA0 86 02              LDA A  #2
0FA2 97 A6              STA A  BYTCNT    SET COUNT
0FA4 08                 INX              MOVE PAST TERM
0FA5 DF 6B              STX    XTEMP1     SET NEW INDEX
0FA7 BD 0B F2           JSR    PEVAL     GO EVALUATE NEXT
0FAA DE 87              LDX    BYTPTR    GET POINTER
0FAC 96 7B              LDA A  QTEMP
0FAE A7 00              STA A  0,X
0FB0 96 7C              LDA A  QTEMP+1
0FB2 A7 01              STA A  1,X
0FB4 08                 INX
0FB5 08                 INX              PUT DATA AND ADJUST
0FB6 DF 87              STX    BYTPTR    SAVE PTR
0FB8 8C 03 00           CPX    #BYTSTK+256
0FBB 20 CD              BRA    TYP10A    LOOP TILL DONE
                *
0FBD 7F 00 59   TYPE11  CLR    PCFLAG    TURN PC OFF
0FC0 96 8F              LDA A  PASS
0FC2 27 25              BEQ    TYP11C    IF PASS 1 IGNORE
0FC4 96 5D              LDA A  P3FLG
0FC6 27 21              BEQ    TYP11C
0FC8 96 4F              LDA A  LABEL
0FCA 26 1E              BNE    TYPERR
0FCC 96 AE              LDA A  LIST      SEE IF LIST ON
0FCE 27 19              BEQ    TYP11C    IF NOT, IGNORE
0FD0 BD 11 D5           JSR    EVAL      CRUNCH IT
0FD3 D6 7C              LDA B  QTEMP+1   GET COUNT
0FD5 26 02              BNE    TYP11A
0FD7 C6 01              LDA B  #1        SET 1 LINE
0FD9 BD 07 BA   TYP11A  JSR    PCRLF     DO LF
0FDC 96 5C              LDA A  EJFLG     SEE IF EJECTED
0FDE 26 03              BNE    TYP11B    IF SO, QUIT
0FE0 5A                 DEC B            COUNT OFF
0FE1 26 F6              BNE    TYP11A    LOOP TILL DONE
0FE3 7F 00 5C   TYP11B  CLR    EJFLG     RESET FLAG
0FE6 7F 00 5E           CLR    PRTFLG    DON'T PRINT
0FE9 39         TYP11C  RTS              DONE
                *
0FEA 7E 10 B4   TYPERR  JMP    TYP15A
                *
0FED 7F 00 59   TYPE12  CLR    PCFLAG
0FF0 96 8F              LDA A  PASS
0FF2 26 F5              BNE    TYP11C
0FF4 96 4F              LDA A  LABEL
0FF6 26 F2              BNE    TYPERR
0FF8 DE 6B      TYP12D  LDX    XTEMP1    GET ARG POINTER
0FFA A6 02              LDA A  2,X
0FFC 97 7D              STA A  TEMP      SAVE
0FFE A6 00              LDA A  0,X
1000 E6 01              LDA B  1,X       GET SWITCH ID
1002 CE 10 41           LDX    #OPTLST   POINT TO LIST
```

```
LOCN B1 B2 B3
1005 A1 00      TYP12A  CMP A  0,X         SEE IF MATCH
1007 27 10              BEQ    TYP12B
1009 08         TYP12C  INX
100A 08                 INX
100B 08                 INX
100C 08                 INX
100D 08                 INX
100E 08                 INX                ADVANCE PTR
100F 8C 10 89           CPX    #OPNEND+6   SEE IF TABLE END
1012 26 F1              BNE    TYP12A      LOOP
1014 86 0A              LDA A  #10
1016 7E 07 E5           JMP    ASMERR      SET ERROR NUMBER AND REPORT
1019 E1 01      TYP12B  CMP B  1,X         SEE IF SECOND MATCH
101B 26 EC              BNE    TYP12C      IF NOT, GO BACK
101D 36                 PSH A
101E 96 7D              LDA A  TEMP        GET 3RD CHAR
1020 A1 02              CMP A  2,X         SEE IF MATCH
1022 32                 PUL A
1023 26 E4              BNE    TYP12C      IF NOT, LOOP
1025 A6 03              LDA A  3,X         GET DATA
1027 EE 04              LDX    4,X         GET ADDRESS
1029 A7 00              STA A  0,X         SET SWITCH
102B DE 6B              LDX    XTEMP1
102D A6 00      FNDEND  LDA A  0,X
102F 08                 INX
1030 DF 6B              STX    XTEMP1
1032 81 0D              CMP A  #$D
1034 27 0A              BEQ    OPTDON
1036 81 20              CMP A  #' 
1038 27 06              BEQ    OPTDON
103A 81 2C              CMP A  #',
103C 27 BA              BEQ    TYP12D
103E 20 ED              BRA    FNDEND
1040 39         OPTDON  RTS                DONE
                *
1041 4C         OPTLST  FCC    'LIS'
1042 49
1043 53
1044 FF                 FCB    $FF
1045 00 AE              FDB    LIST
1047 4E                 FCC    'NOL'
1048 4F
1049 4C
104A 00                 FCB    0
104B 00 AE              FDB    LIST
104D 54                 FCC    'TAP'
104E 41
104F 50
1050 FF                 FCB    $FF
1051 00 B2              FDB    TAPE
1053 4E                 FCC    'NOT'
1054 4F
1055 54
1056 00                 FCB    0
1057 00 B2              FDB    TAPE
```

```
LOCN B1 B2 B3
1059 4D                    FCC     'MEM'
105A 45
105B 4D
105C FF                    FCB     $FF
105D 00 B3                 FDB     MEMORY
105F 4E                    FCC     'NOM'
1060 4F
1061 4D
1062 00                    FCB     0
1063 00 B3                 FDB     MEMORY
1065 53                    FCC     'SYM'
1066 59
1067 4D
1068 FF                    FCB     $FF
1069 00 AF                 FDB     SYMBOL
106B 4E                    FCC     'NOS'
106C 4F
106D 53
106E 00                    FCB     0
106F 00 AF                 FDB     SYMBOL
1071 47                    FCC     'GEN'
1072 45
1073 4E
1074 FF                    FCB     $FF
1075 00 B0                 FDB     GENER
1077 4E                    FCC     'NOG'
1078 4F
1079 47
107A 00                    FCB     0
107B 00 B0                 FDB     GENER
107D 50                    FCC     'PAG'
107E 41
107F 47
1080 FF                    FCB     $FF
1081 00 B1                 FDB     PAGER
1083 4E          OPNEND    FCC     'NOP'
1084 4F
1085 50
1086 00                    FCB     0
1087 00 B1                 FDB     PAGER
                 *
                 *
1089 7F 00 59    TYPE13    CLR     PCFLAG
108C 96 8F                 LDA A   PASS
108E 27 11                 BEQ     TYP13A
1090 96 4F                 LDA A   LABEL
1092 26 20                 BNE     TYP15A
1094 97 5E                 STA A   PRTFLG
1096 96 B1                 LDA A   PAGER      SEE IF PAGER ON
1098 27 07                 BEQ     TYP13A     IF NOT, IGNORE
109A 96 AE                 LDA A   LIST       SEE IF LIST ON
109C 27 03                 BEQ     TYP13A     IF NOT, IGNORE
109E 7F 00 5F              CLR     PAGFLG
10A1 39          TYP13A    RTS
                 *
```

```
LOCN B1 B2 B3
10A2 96 4F      TYPE14  LDA A  LABEL
10A4 26 0E              BNE    TYP15A
10A6 BD 11 D5           JSR    EVAL        GO EVALUATE OPERAND
10A9 DE 7B              LDX    QTEMP       GET RESULT
10AB DF 4B              STX    PC          SET PC
10AD DF 6D              STX    XTEMP2
10AF 39                 RTS
                *
10B0 96 4F      TYPE15  LDA A  LABEL
10B2 26 05              BNE    EQU1
10B4 86 07      TYP15A  LDA A  #7           SET ERROR
10B6 7E 07 E5           JMP    ASMERR
10B9 BD 09 05   EQU1    JSR    FNDLBL      FIND LABEL
10BC DF FD              STX    $FD
10BE 96 8F              LDA A  PASS        CHECK PASS
10C0 4A                 DEC A
10C1 97 56              STA A  ERRFLG
10C3 BD 11 D5           JSR    EVAL        GO EVALUATE
10C6 7F 00 56           CLR    ERRFLG
10C9 DE FD              LDX    $FD
10CB 96 7C              LDA A  QTEMP+1
10CD D6 7B              LDA B  QTEMP
10CF E7 06              STA B  6,X
10D1 A7 07              STA A  7,X
10D3 DE 7B              LDX    QTEMP
10D5 DF 6D              STX    XTEMP2
10D7 39         TYP15C  RTS
10D8 96 84              LDA A  LSTERR      ELSE ERROR
10DA 7E 07 E5           JMP    ASMERR      GO REPORT
                *
10DD 7F 00 59   TYPE16  CLR    PCFLAG
10E0 96 4F              LDA A  LABEL
10E2 26 D0              BNE    TYP15A
10E4 86 FF              LDA A  #$FF
10E6 97 58              STA A  ENDFLG
10E8 39                 RTS
                *
10E9 7F 00 59   TYPE17  CLR    PCFLAG
10EC 96 8F              LDA A  PASS
10EE 27 2E              BEQ    NAM3        IF PASS1 IGNORE
10F0 96 4F              LDA A  LABEL
10F2 26 C0              BNE    TYP15A
10F4 CE 00 C6           LDX    #TITLE
10F7 DF 65              STX    XSAVE       SAVE PTR
10F9 DE 96      NAM1    LDX    OPNPTR      GET POINTER
10FB A6 00              LDA A  0,X
10FD 81 0D              CMP A  #$D         CHECK FOR CR
10FF 27 0F              BEQ    NAM2
1101 08                 INX    GET TO NEXT
1102 DF 96              STX    OPNPTR
1104 DE 65              LDX    XSAVE       GET OTHER PTR
1106 A7 00              STA A  0,X
1108 08                 INX
1109 DF 65              STX    XSAVE       UPDATE
110B 8C 00 E6           CPX    #TITLE+32
```

```
LOCN B1 B2 B3
110E 26 E9          BNE    NAM1
1110 86 20    NAM2  LDA A  #$20
1112 DE 65          LDX    XSAVE
1114 8C 00 E6 FILTIT CPX   #TITLE+32
1117 27 05          BEQ    NAM3
1119 A7 00          STA A  0,X
111B 08             INX
111C 20 F6          BRA    FILTIT
111E 39       NAM3  RTS
              *
111F BD 11 D5 TYPE18 JSR   EVAL
1122 CE 00 7B       LDX    #QTEMP
1125 D6 4C          LDA B  PC+1
1127 96 4B          LDA A  PC
1129 BD 0C FA       JSR    ADD16
112C DE 7B          LDX    QTEMP
112E DF 4B          STX    PC
1130 39             RTS
              *
              ** EJECT
1131 37       EJECT PSH B
1132 D6 B1          LDA B  PAGER      SEE IF PAGE ON
1134 27 65          BEQ    NOEJT      IF NOT, SKIP
1136 CE 11 D1       LDX    #EJSTR     POINT TO EJECT STRING
1139 BD 07 AB       JSR    PDATA      PRINT THE CHARS
113C 37             PSH B
113D 4F             CLR A
113E 97 A8          STA A  LINCNT
1140 97 B1          STA A  PAGER      TURN PAGER OFF
1142 C6 03          LDA B  #3
1144 27 06          BEQ    MARDON
1146 BD 07 BA PRTMAR JSR   PCRLF
1149 5A             DEC B
114A 26 FA          BNE    PRTMAR     PRINT MARGIN
114C CE 00 C6 MARDON LDX   #TITLE
114F BD 07 AB       JSR    PDATA      SET IN TITLE
1152 CE 11 A9       LDX    #PPP
1155 BD 07 AB       JSR    PDATA      PRINT HEADER
1158 96 AD          LDA A  PAGENO+1
115A 8B 01          ADD A  #1       KICK PAGE COUNT
115C 19             DAA
115D 97 AD          STA A  PAGENO+1
115F 96 AC          LDA A  PAGENO
1161 89 00          ADC A  #0
1163 19             DAA
1164 97 AC          STA A  PAGENO
1166 27 0C          BEQ    PPAG2
1168 84 F0          AND A  #$F0
116A 27 03          BEQ    PPAG6
116C 8D 2F          BSR    OUTHL      PRINT MS
116E 5C             INC B              SET FLAG
116F 96 AC    PPAG6 LDA A  PAGENO     GET BYTE
1171 8D 30          BSR    OUTHR      PRINT LS OF MS
1173 5C             INC B
1174 96 AD    PPAG2 LDA A  PAGENO+1   GET LS BYTE
```

```
LOCN B1 B2 B3
1176 27 1E              BEQ    PPAG3
1178 5D                 TST B                SEE IF PRINTED YET
1179 26 04              BNE    PPAG5         IF SO, JUST PRINT
117B 85 F0              BIT A  #$F0          CHECK MS DIGIT
117D 27 04              BEQ    PPAG4         IF 0, DON'T PRINT
117F 8D 1C      PPAG5   BSR    OUTHL         PRINT
1181 96 AD              LDA A  PAGENO+1
1183 8D 1E      PPAG4   BSR    OUTHR
1185 BD 07 BA           JSR    PCRLF
1188 BD 07 BA           JSR    PCRLF
118B 86 FF              LDA A  #$FF
118D 97 5C              STA A  EJFLG
118F 97 5F              STA A  PAGFLG
1191 33                 PUL B                GET PAGE STATUS
1192 D7 B1              STA B  PAGER         RESTORE
1194 33                 PUL B
1195 39                 RTS                  DONE
1196 5D         PPAG3   TST B                CHECK IF PRINTED
1197 26 E6              BNE    PPAG5
1199 20 E8              BRA    PPAG4
119B 33         NOEJT   PUL B
119C 39                 RTS                  DONE
119D BD 0C DB   OUTHL   JSR    HEXL
11A0 7E 03 20           JMP    OUTCH
11A3 BD 0C DF   OUTHR   JSR    HEXR
11A6 7E 03 20           JMP    OUTCH
11A9 20         PPP     FCC    '          '
11AA 20
11AB 20
11AC 20
11AD 20
11AE 20
11AF 20
11B0 20
11B1 54                 FCC    'TSC MNEMONIC ASSEMBLER    PAGE '
11B2 53
11B3 43
11B4 20
11B5 4D
11B6 4E
11B7 45
11B8 4D
11B9 4F
11BA 4E
11BB 49
11BC 43
11BD 20
11BE 41
11BF 53
11C0 53
11C1 45
11C2 4D
11C3 42
11C4 4C
11C5 45
```

```
LOCN B1 B2 B3
11C6 52
11C7 20
11C8 20
11C9 20
11CA 20
11CB 50
11CC 41
11CD 47
11CE 45
11CF 20
11D0 04                 FCB     4
11D1 00           EJSTR FCB     0,0,$A,4
11D2 00
11D3 0A
11D4 04
                  *
                  ** EVAL
                  * EVALUATE AN OPERAND EXPRESSION
11D5 4F           EVAL  CLR A
11D6 97 7B              STA A   QTEMP
11D8 97 7C              STA A   QTEMP+1
11DA 97 63              STA A   OPN         INITIALIZE
11DC DE 6B              LDX     XTEMP1
11DE DF 96              STX     OPNPTR      SET POINTER
11E0 DE 96        EVAL1A LDX    OPNPTR      GET OPERAND PTR
11E2 A6 00        FINDSC LDA A  0,X         GET CHAR
11E4 08                 INX
11E5 5F                 CLR B
11E6 81 2B              CMP A   #'+
11E8 27 27              BEQ     F1
11EA 5C                 INC B
11EB 81 2D              CMP A   #'-
11ED 27 22              BEQ     F1
11EF 5C                 INC B
11F0 81 2A              CMP A   #'*
11F2 26 0A              BNE     FINDS4
11F4 09                 DEX
11F5 9C 96              CPX     OPNPTR
11F7 07                 TPA
11F8 08                 INX
11F9 06                 TAP
11FA 27 E6              BEQ     FINDSC
11FC 20 13              BRA     F1
11FE 5C           FINDS4 INC B
11FF 81 2F              CMP A   #'/
1201 27 0E              BEQ     F1
1203 C6 FF        F2    LDA B   #$FF
1205 81 20              CMP A   #'
1207 27 08              BEQ     F1
1209 81 2C              CMP A   #',
120B 27 04              BEQ     F1
120D 81 0D              CMP A   #$D
120F 26 D1              BNE     FINDSC
1211 D7 64        F1    STA B   TERM        SAVE TERMINATOR
1213 09                 DEX                 ADJUST
```

```
LOCN B1 B2 B3
1214 DF 6B          STX    XTEMP1
1216 DE 96    LOAD  LDX    OPNPTR    GET POINTER
1218 7F 00 7D       CLR    TEMP
121B A6 00          LDA A  0,X       GET CHARACTER
121D 81 41          CMP A  #'A
121F 25 1F          BCS    LOAD1
1221 81 5A          CMP A  #'Z
1223 22 1B          BHI    LOAD1     CHECK FOR LABEL
1225 DF 79          STX    QTEMP2    SAVE X
1227 BD 0C 65       JSR    CLRLAB    SET LABEL TO ZERO
122A DE 79          LDX    QTEMP2    GET X BACK
122C BD 0C 8F       JSR    COPLBL
122F BD 09 05       JSR    FNDLBL    GO GET VALUE
1232 EE 06          LDX    6,X       GET VALUE
1234 DF 79          STX    QTEMP2    STORE IT
1236 DE 6B          LDX    XTEMP1
1238 4D             TST A             SEE IF FOUND
1239 2A 50          BPL    L5
123B 86 01          LDA A  #1
123D 7E 12 98       JMP    F3
1240 C6 01    LOAD1 LDA B  #1         SET ID
1242 81 24          CMP A  #'$        CHECK FOR BASE TAGS
1244 27 2F          BEQ    L1
1246 5C             INC B
1247 81 25          CMP A  #$25       PERCENT
1249 27 2A          BEQ    L1
124B 5C             INC B
124C 81 40          CMP A  #'@
124E 27 25          BEQ    L1
1250 5C             INC B
1251 81 27          CMP A  #$27       CHECK FOR SINGLE QUOTE
1253 27 20          BEQ    L1
1255 DE 6B          LDX    XTEMP1     GET END POINTER
1257 09             DEX               MOVE TO LAST CHAR
1258 7C 00 7D       INC    TEMP
125B 5A             DEC B
125C A6 00          LDA A  0,X        GET IT
125E 81 4F          CMP A  #'O        CHECK OCTAL
1260 27 16          BEQ    L2
1262 81 51          CMP A  #'Q        CHECK OCTAL
1264 27 12          BEQ    L2
1266 5A             DEC B
1267 81 42          CMP A  #'B        CHECK BINARY
1269 27 0D          BEQ    L2
126B 5A             DEC B
126C 81 48          CMP A  #'H        CHECK HEX
126E 27 08          BEQ    L2
1270 5A             DEC B             SET DECIMAL
1271 D7 7D          STA B  TEMP
1273 20 03          BRA    L2
1275 08       L1    INX               MOVE TO FIRST CHAR OF CONST
1276 DF 96          STX    OPNPTR     SAVE
1278 4F       L2    CLR A
1279 97 79          STA A  QTEMP2
127B 97 7A          STA A  QTEMP2+1
```

```
LOCN B1 B2 B3
127D CE 12 C9          LDX    #BCONV    POINT TO TABLE
1280 58                ASL B
1281 27 04             BEQ    L4
1283 08          L3    INX
1284 5A                DEC B
1285 26 FC             BNE    L3        GET TO ADDRESS
1287 EE 00       L4    LDX    0,X       GET ADDRESS
1289 AD 00             JSR    0,X       COLLECT DATA
128B 96 7D       L5    LDA A  TEMP      CHECK PRE OR POST
128D 27 01             BEQ    L6
128F 08                INX
1290 DF 71       L6    STX    XTEMP4    SAVE
1292 9C 6B             CPX    XTEMP1    SEE IF GOT ALL
1294 27 0B             BEQ    EVAL1B
1296 86 09             LDA A  #9
1298 7F 00 7B    F3    CLR    QTEMP
129B 7F 00 7C          CLR    QTEMP+1   RESET ARG
129E 7E 07 E5          JMP    ASMERR    GO TO ERROR
12A1 96 63       EVAL1B LDA A  OPN       GET OPERATION
12A3 CE 12 C1          LDX    #OPNTBL   POINT TO JUMP TABLE
12A6 48                ASL A
12A7 27 04             BEQ    EVAL3
12A9 08          EVAL2 INX               POINT NEXT
12AA 4A                DEC A
12AB 26 FC             BNE    EVAL2     MOVE TO TARGET
12AD EE 00       EVAL3 LDX    0,X       GET TARGET ADDR.
12AF AD 00             JSR    0,X       DO OPERATION
12B1 DE 6B             LDX    XTEMP1    GET POINTER
12B3 08                INX
12B4 DF 96             STX    OPNPTR    SAVE PLACE
12B6 96 64             LDA A  TERM      GET LAST TERM
12B8 97 63             STA A  OPN       SAVE OPERATION
12BA 2B 03             BMI    EVAL4     IF A TERMINATOR, DONE
12BC 7E 11 E0          JMP    EVAL1A    ELSE PROCESS AGAIN
12BF 4F          EVAL4 CLR A             DONE
12C0 39                RTS
                 *
12C1 12 D3       OPNTBL FDB    OPADD
12C3 12 DD             FDB    OPSUB
12C5 12 E7             FDB    OPMUL
12C7 13 0F             FDB    OPDIV
                 *
12C9 13 5B       BCONV FDB     DECM
12CB 13 9A             FDB    HEX
12CD 13 BA             FDB    BIN
12CF 13 D0             FDB    OCT
12D1 13 E7             FDB    ASC
                 *
12D3 96 79       OPADD LDA A   QTEMP2
12D5 D6 7A             LDA B   QTEMP2+1  GET OPERAND
12D7 CE 00 7B          LDX    #QTEMP     POINT TO ACC.
12DA 7E 0C FA          JMP    ADD16      GO ADD
                 *
12DD 96 79       OPSUB LDA A   QTEMP2
12DF D6 7A             LDA B   QTEMP2+1
```

```
LOCN B1 B2 B3
12E1 CE 00 7B          LDX     #QTEMP
12E4 7E 0C E8          JMP     SUB16
                       *
12E7 CE 00 00  OPMUL   LDX     #0
12EA DF 77             STX     QTEMP3      SET ACCUM.
12EC CE 00 77          LDX     #QTEMP3
12EF C6 10             LDA B   #16         SET COUNT
12F1 A6 03     OPMUL2  LDA A   3,X
12F3 46                ROR A               CHECK BIT
12F4 24 09             BCC     OPMUL3
12F6 37                PSH B
12F7 A6 04             LDA A   4,X
12F9 E6 05             LDA B   5,X         GET OPERANDS
12FB BD 0C FA          JSR     ADD16       ADD IN
12FE 33                PUL B
12FF 64 00     OPMUL3  LSR     0,X
1301 66 01             ROR     1,X
1303 66 02             ROR     2,X
1305 66 03             ROR     3,X
1307 5A                DEC B               COUNT OFF
1308 26 E7             BNE     OPMUL2
130A EE 02             LDX     2,X         GET RESULT
130C DF 7B             STX     QTEMP       SAVE
130E 39                RTS
                       *
130F CE 00 00  OPDIV   LDX     #0
1312 DF 77             STX     QTEMP3      INIT. ACCUM.
1314 DE 79             LDX     QTEMP2
1316 D6 7C             LDA B   QTEMP+1
1318 D7 7A             STA B   QTEMP2+1
131A D6 7B             LDA B   QTEMP
131C D7 79             STA B   QTEMP2
131E DF 7B             STX     QTEMP       MOVE OPERAND
1320 C6 11             LDA B   #17         SET COUNT
1322 CE 00 77          LDX     #QTEMP3     POINT TO ACC.
1325 37        OPDIV1  PSH B
1326 96 7B             LDA A   QTEMP
1328 D6 7C             LDA B   QTEMP+1
132A BD 0C E8          JSR     SUB16
132D 25 08             BCS     OPDIV3
132F 96 7B             LDA A   QTEMP
1331 D6 7C             LDA B   QTEMP+1
1333 BD 0C FA          JSR     ADD16       ADD BACK
1336 0C                CLC
1337 69 03     OPDIV3  ROL     3,X
1339 69 02             ROL     2,X
133B 69 01             ROL     1,X
133D 69 00             ROL     0,X         SHIFT IT
133F 33                PUL B               RETRIEVE COUNT
1340 5A                DEC B               COUNT OFF
1341 26 E2             BNE     OPDIV1      DO AGAIN
1343 EE 02             LDX     2,X         GET RESULT
1345 DF 7B             STX     QTEMP       SAVE
1347 39                RTS                 DONE
                       *
```

```
LOCN B1 B2 B3
                    *
1348 E6 00          INDEC   LDA B   0,X         GET A CHAR
134A C0 3A                  SUB B   #$3A        REMOVE BIAS
134C 24 02                  BCC     INDEC2
134E CB 0A                  ADD B   #$A         CORRECT
1350 39             INDEC2  RTS
                    *
1351 96 6D          SPCL    LDA A   XTEMP2
1353 97 79                  STA A   QTEMP2
1355 96 6E                  LDA A   XTEMP2+1
1357 97 7A                  STA A   QTEMP2+1
1359 08                     INX                 ALIGN POINTER
135A 39                     RTS
                    *
135B 8D 2B          DECM    BSR     INITR       GO INITIALIZE
135D A6 00                  LDA A   0,X
135F 81 2A                  CMP A   #'*         CHECK SPECIAL CHAR
1361 27 EE                  BEQ     SPCL
1363 8D E3          DECM2   BSR     INDEC       GO FETCH
1365 24 20                  BCC     DECM3
1367 37                     PSH B
1368 96 79                  LDA A   QTEMP2
136A D6 7A                  LDA B   QTEMP2+1
136C 8D 25                  BSR     LONE        LEFT ONE
136E 8D 23                  BSR     LONE        AGAIN
1370 DB 7A                  ADD B   QTEMP2+1
1372 D7 7A                  STA B   QTEMP2+1    ADD IN
1374 99 79                  ADC A   QTEMP2
1376 97 79                  STA A   QTEMP2
1378 8D 19                  BSR     LONE        LEFT AGAIN
137A 33                     PUL B
137B 4F                     CLR A
137C DB 7A                  ADD B   QTEMP2+1
137E 99 79                  ADC A   QTEMP2
1380 D7 7A                  STA B   QTEMP2+1
1382 97 79                  STA A   QTEMP2
1384 08                     INX
1385 20 DC                  BRA     DECM2       GO AT IT AGAIN
1387 39             DECM3   RTS
                    *
1388 DE 96          INITR   LDX     OPNPTR      GET POINTER
138A 7F 00 79               CLR     QTEMP2
138D 7F 00 7A               CLR     QTEMP2+1    ZERO ACCUMULATOR
1390 39                     RTS
                    *
1391 8D 00          LTWO    BSR     LONE        LEFT ONE
                    *
1393 78 00 7A       LONE    ASL     QTEMP2+1
1396 79 00 79               ROL     QTEMP2
1399 39                     RTS
                    *
139A 8D EC          HEX     BSR     INITR       INITIALIZE
139C A6 00          HEX2    LDA A   0,X         GET CHAR
139E 80 47                  SUB A   #'G         REMOVE BIAS
13A0 2A 17                  BPL     HEX4
```

```
LOCN B1 B2 B3
13A2 8B 06          ADD A  #6          ADD ON
13A4 2A 04          BPL    HEX3
13A6 8B 07          ADD A  #7          ADD AGAIN
13A8 2A 0F          BPL    HEX4        REMOVE $3A - $40
13AA 8B 0A    HEX3  ADD A  #10         CORRECT
13AC 2B 0B          BMI    HEX4        REMOVE <$30
13AE 8D E1          BSR    LTWO
13B0 8D DF          BSR    LTWO
13B2 9B 7A          ADD A  QTEMP2+1
13B4 97 7A          STA A  QTEMP2+1
13B6 08             INX
13B7 20 E3          BRA    HEX2
13B9 39       HEX4  RTS
              *
13BA 8D CC    BIN   BSR    INITR
13BC A6 00    BIN2  LDA A  0,X
13BE 80 30          SUB A  #$30
13C0 2B F7          BMI    HEX4
13C2 81 01          CMP A  #1
13C4 22 F3          BHI    HEX4
13C6 46             ROR A
13C7 79 00 7A       ROL    QTEMP2+1
13CA 79 00 79       ROL    QTEMP2
13CD 08             INX
13CE 20 EC          BRA    BIN2
              *
13D0 8D B6    OCT   BSR    INITR
13D2 A6 00    OCT1  LDA A  0,X
13D4 80 30          SUB A  #$30
13D6 2B E1          BMI    HEX4
13D8 81 07          CMP A  #7
13DA 22 DD          BHI    HEX4
13DC 8D B3          BSR    LTWO
13DE 8D B3          BSR    LONE        MULT X 8
13E0 9B 7A          ADD A  QTEMP2+1
13E2 97 7A          STA A  QTEMP2+1
13E4 08             INX
13E5 20 EB          BRA    OCT1
13E7 8D 9F    ASC   BSR    INITR       GO INITIALIZE
13E9 A6 00          LDA A  0,X         GET CHAR
13EB 97 7A          STA A  QTEMP2+1    SET CHAR
13ED DE 6B          LDX    XTEMP1      IGNORE REST
13EF 39             RTS                DONE
              *
              *
              ** SHELL
              * DO A SHELL SORT
13F0 7F 00 7D SHELL CLR    TEMP
13F3 86 08          LDA A  #8
13F5 36             PSH A
13F6 86 20          LDA A  #32
13F8 36             PSH A
13F9 86 68          LDA A  #104
13FB 36             PSH A              SET GAP WIDTHS
13FC 32       SHELL1 PUL A           GET A GAP
```

```
LOCN B1 B2 B3
13FD 97 AA          STA A  GAP        SAVE
13FF DE 40          LDX    LBLBEG
1401 DF 77    SHELL2 STX   QTEMP3     SAVE PLACE
1403 DF 7B    SETGAP STX   QTEMP      SAVE PTR
1405 96 7C          LDA A  QTEMP+1
1407 9B AA          ADD A  GAP
1409 97 7A          STA A  QTEMP2+1
140B 96 7B          LDA A  QTEMP
140D 89 00          ADC A  #0
140F 97 79          STA A  QTEMP2     SET BOTTOM POINTER
1411 91 42          CMP A  LBLEND
1413 25 08          BCS    SORT
1415 26 60          BNE    PASDON
1417 96 7A          LDA A  QTEMP2+1
1419 91 43          CMP A  LBLEND+1
141B 24 5A          BCC    PASDON
141D C6 06    SORT  LDA B  #6         SET FOR 6 COMPARES
141F DE 7B          LDX    QTEMP      GET TOP PTR
1421 DF 69          STX    XTEMP      SAVE
1423 DE 79          LDX    QTEMP2     GET BOTTOM PTR
1425 DF 6D          STX    XTEMP2     SAVE
1427 DE 69    CHKLOP LDX   XTEMP      GET PTR
1429 A6 00          LDA A  0,X
142B 08             INX                GET CHAR AND ADV.
142C DF 69          STX    XTEMP
142E DE 6D          LDX    XTEMP2     GET PTR
1430 A1 00          CMP A  0,X        CHECK RELATION
1432 27 4D          BEQ    SAME       SAME?
1434 23 30          BLS    ORDOK      IN ORDER?
1436 C6 08          LDA B  #8         SET 8 TRANSFERS
1438 DE 7B          LDX    QTEMP      GET TABLE PTR
143A DF 69    MOVELP STX   XTEMP      SAVE
143C 37             PSH B              SAVE COUNT
143D A6 00          LDA A  0,X
143F DE 79          LDX    QTEMP2     GET DEST PTR
1441 E6 00          LDA B  0,X
1443 A7 00          STA A  0,X        SWITCH
1445 08             INX
1446 DF 79          STX    QTEMP2     SAVE PTR
1448 DE 69          LDX    XTEMP      GET DEST PTR
144A E7 00          STA B  0,X        SWITCH
144C 08             INX
144D 33             PUL B
144E 5A             DEC B
144F 26 E9          BNE    MOVELP     LOOP TILL DONE
1451 96 7D          LDA A  TEMP       GET FLAG
1453 26 03          BNE    SHELL5
1455 73 00 7D       COM    TEMP       CHANGE FLAG
1458 DE 7B    SHELL5 LDX   QTEMP      GET PTR
145A 9C 40          CPX    LBLBEG     SEE IF AT TOP
145C 27 08          BEQ    ORDOK      IF SO, GO DOWN
145E C6 08          LDA B  #8
1460 09       DECXX DEX                MOVE BACK
1461 5A             DEC B
1462 26 FC          BNE    DECXX
```

```
LOCN B1 B2 B3
1464 20 9D          BRA     SETGAP
1466 96 7D    ORDOK LDA A   TEMP      GET FLAG
1468 27 03          BEQ     SHELL6    IF 0, FOWARD
146A 7F 00 7D       CLR     TEMP      SET FOWARD
146D DE 77    SHELL6 LDX    QTEMP3    GET LIST POINTER
146F C6 08          LDA B   #8        SET FOR NEXT
1471 08       OFFLOP INX
1472 5A             DEC B             MOVE PTR
1473 26 FC          BNE     OFFLOP
1475 20 8A          BRA     SHELL2
1477 96 AA    PASDON LDA A  GAP       GET DISTANCE
1479 81 08          CMP A   #8
147B 27 03          BEQ     SRTDON    IF 8, DONE
147D 7E 13 FC       JMP     SHELL1
1480 39       SRTDON RTS
1481 08       SAME  INX
1482 DF 6D          STX     XTEMP2    SAVE PTR
1484 5A             DEC B             CHECKED ALL 6?
1485 26 A0          BNE     CHKLOP
1487 20 DD          BRA     ORDOK
                    *
                    *
                    ** OBJCOD
                    * PRODUCE MIKBUG RECORD FORMAT
1489 96 62    OBJCOD LDA A  OBJINT    SEE IF FIRST CALL
148B 27 0C          BEQ     OBJC01    IF SO, SKIP
148D CE 04 C0       LDX     #TAPEON
1490 BD 04 B2       JSR     CONTRL    TURN TAPE ON
1493 BD 04 C8       JSR     DELAY     DELAY FOR STARTUP
1496 7F 00 62       CLR     OBJINT    RESET FLAG
1499 DE 6D    OBJC01 LDX    XTEMP2    GET PC (LAST TIME'S)
149B 9C 9E          CPX     LASTPC
149D 07             TPA
149E DE 4B          LDX     PC
14A0 DF 9E          STX     LASTPC    SET NEW LAST PC
14A2 06             TAP               RESTORE CCR
14A3 27 03          BEQ     OBJC04    SEE IF NEW ORG
14A5 BD 15 18       JSR     PRTREC    IF SO, PRINT LAST PART
14A8 96 90    OBJC04 LDA A  OPCNT     GET BYTE COUNTER
14AA D6 A7    OBJC03 LDA B  BUFCNT    GET BUFFER COUNT
14AC 26 04          BNE     OBJC05    IF NOT EMPTY, SKIP
14AE DE 6D          LDX     XTEMP2    GET PC
14B0 DF A0          STX     OBJADR    SET RECORD ADDRESS
14B2 DE 89    OBJC05 LDX    OBJPTR    GET DEST PTR
14B4 D6 7E          LDA B   OPCODE
14B6 E7 00          STA B   0,X
14B8 08             INX
14B9 7C 00 A7       INC     BUFCNT
14BC 4A             DEC A
14BD 27 13          BEQ     OBJC06
14BF D6 7F          LDA B   OP1
14C1 E7 00          STA B   0,X
14C3 08             INX
14C4 7C 00 A7       INC     BUFCNT
14C7 4A             DEC A
```

```
LOCN B1 B2 B3
14C8 27 08          BEQ    OBJC06
14CA D6 80          LDA B  OP2
14CC E7 00          STA B  0,X
14CE 08             INX
14CF 7C 00 A7       INC    BUFCNT     PUT DATA, SET COUNT
14D2 8D 20    OBJC06 BSR   CHKGEN     GO CHECK IF BUF. FULL
14D4 96 5A          LDA A  DATFLG     CHECK FCC,FCB,FDB
14D6 27 3F          BEQ    OBJDON     IF NOT, DONE
14D8 CE 02 00       LDX    #BYTSTK
14DB DF 71          STX    XTEMP4     SET DATA BUFFER POINTER
14DD DE 71    OBJC07 LDX   XTEMP4     GET DATA POINTER
14DF 9C 87          CPX    BYTPTR     SEE IF EMPTY
14E1 27 34          BEQ    OBJDON     IF SO, DONE
14E3 A6 00          LDA A  0,X        GET DATA
14E5 08             INX
14E6 DF 71          STX    XTEMP4     FIX PTR
14E8 DE 89          LDX    OBJPTR     GET PTR
14EA A7 00          STA A  0,X        PUT DATA
14EC 08             INX                ADVANCE
14ED 7C 00 A7       INC    BUFCNT     FIX COUNT
14F0 8D 02          BSR    CHKGEN     CHECK GENERATE TIME
14F2 20 E9          BRA    OBJC07     LOOP TILL EMPTY
14F4 DF 89    CHKGEN STX   OBJPTR     SAVE POINTER
14F6 96 A7          LDA A  BUFCNT     GET COUNT
14F8 81 0F          CMP A  #15
14FA 22 01          BHI    GENOBJ     IF >=16  TIME TO PUNCH
14FC 39             RTS
14FD 36       GENOBJ PSH A              SAVE COUNT
14FE 86 10          LDA A  #16        SET BYTE COUNT
1500 BD 15 1C       JSR    RECORD     GO PUNCH RECORD
1503 32             PUL A             GET COUNT
1504 CE 00 B4       LDX    #OBJBUF
1507 80 10          SUB A  #16        CALCULATE DATA LEFT
1509 97 A7          STA A  BUFCNT     UPDATE COUNT
150B 27 08    SHIFTL BEQ   SAVEPL     IF 0, HAVE PLACE
150D E6 10    MOVE   LDA B 16,X       GET DATA
150F E7 00          STA B  0,X
1511 08             INX               MOVE PTR
1512 4A             DEC A             KICK COUNT
1513 26 F8          BNE    MOVE       MOVE ALL DATA
1515 DF 89    SAVEPL STX   OBJPTR     SAVE BUFFER PTR
1517 39       OBJDON RTS              DONE
1518 96 A7    PRTREC LDA A BUFCNT     GET  COUNT
151A 27 FB          BEQ    OBJDON     IF 0, NOTHING TO PUNCH
151C 36       RECORD PSH A            SAVE COUNT
151D 7F 00 A7       CLR    BUFCNT     SET COUNT 0
1520 CE 00 B4       LDX    #OBJBUF
1523 DF 89          STX    OBJPTR     RESET POINTER
1525 8D 3D          BSR    HEADER     PUNCH HEADER
1527 32             PUL A
1528 36             PSH A             GET COUNT
1529 8B 03          ADD A  #3         SET BYTE COUNT
152B 8D 23          BSR    TAPBYT     PUNCH BYTE
152D 96 A0          LDA A  OBJADR     GET MS ADDRESS
152F BD 15 50       JSR    TAPBYT
```

```
LOCN B1 B2 B3
1532 96 A1              LDA A   OBJADR+1
1534 8D 1A              BSR     TAPBYT
1536 32                 PUL A
1537 36                 PSH A                   GET COUNT AGAIN
1538 9B A1              ADD A   OBJADR+1
153A 97 A1              STA A   OBJADR+1
153C 96 A0              LDA A   OBJADR
153E 89 00              ADC A   #0
1540 97 A0              STA A   OBJADR          SET NEW ADDRESS
1542 33                 PUL B                   GET COUNT
1543 DE 89              LDX     OBJPTR
1545 A6 00      OBJLP   LDA A   0,X             GET DATA
1547 8D 07              BSR     TAPBYT          PUNCH IT
1549 08                 INX
154A 5A                 DEC B                   CHECK DONE
154B 26 F8              BNE     OBJLP
154D 96 61              LDA A   CKSUM           GET CHECKSUM
154F 43                 COM A                   CORRECT
                *
                ** TAPBYT
                * PUNCH A BYTE AND CALC CHECKSUM
1550 36         TAPBYT  PSH A                   SAVE BYTE
1551 9B 61              ADD A   CKSUM           UPDATE CHECKSUM
1553 97 61              STA A   CKSUM
1555 32                 PUL A
1556 36                 PSH A                   GET CHAR
1557 BD 0C DB           JSR     HEXL
155A BD 03 23           JSR     TAPOUT
155D 32                 PUL A
155E BD 0C DF           JSR     HEXR
1561 7E 03 23           JMP     TAPOUT
                *
1564 CE 15 6F   HEADER  LDX     #LNHDX
1567 C6 08              LDA B   #8
1569 7F 00 61           CLR     CKSUM           SET CHECKSUM
156C 7E 04 B6           JMP     PCTRL           GO PUNCH
156F 0D         LNHDX   FCB     $D,$A,0,0,0,0
1570 0A
1571 00
1572 00
1573 00
1574 00
1575 53                 FCC     'S1'
1576 31
                *
                *
                *
                *
                ** MEMCOD
                * INSTALL OBJECT CODE IN MEMORY
1577 DE 6D      MEMCOD  LDX     XTEMP2          GET PC
1579 9C 9C              CPX     LSTPCM          CHECK CONTIGUOUS CODE
157B 07                 TPA
157C DE 4B              LDX     PC
157E DF 9C              STX     LSTPCM
```

```
LOCN B1 B2 B3
1580 06                  TAP              RESTORE STATUS
1581 27 20               BEQ    MEM2      IF CONT., SKIP
1583 DE 8B               LDX    MEMPTR    GET POINTER
1585 96 6D               LDA A  XTEMP2    GET PC
1587 A7 02               STA A  2,X
1589 96 6E               LDA A  XTEMP2+1
158B A7 03               STA A  3,X       PUT IN MEMORY
158D 9C 49               CPX    MEMOBJ    CHECK BEGINNING
158F 27 03               BEQ    MEM1
1591 BD 15 F4            JSR    FIXCNT    GO FIX BYTE COUNT
1594 DE 8B      MEM1     LDX    MEMPTR    GET POINTER
1596 DF A2               STX    LASTM     SAVE PLACE
1598 08                  INX
1599 08                  INX
159A 08                  INX
159B 08                  INX
159C 4F                  CLR A
159D 97 9A               STA A  MCOUNT
159F 97 9B               STA A  MCOUNT+1  SET BYTE COUNT
15A1 DF 8B               STX    MEMPTR    SAVE PTR
15A3 DE 8B      MEM2     LDX    MEMPTR    GET POINTER
15A5 D6 90               LDA B  OPCNT     GET COUNT
15A7 96 7E               LDA A  OPCODE
15A9 A7 00               STA A  0,X
15AB 08                  INX
15AC BD 15 E7            JSR    INCCNT
15AF 5A                  DEC B
15B0 27 13               BEQ    MEM3
15B2 96 7F               LDA A  OP1
15B4 A7 00               STA A  0,X
15B6 08                  INX
15B7 BD 15 E7            JSR    INCCNT
15BA 5A                  DEC B
15BB 27 08               BEQ    MEM3
15BD 96 80               LDA A  OP2
15BF A7 00               STA A  0,X
15C1 08                  INX
15C2 BD 15 E7            JSR    INCCNT
15C5 DF 8B      MEM3     STX    MEMPTR    SAVE PLACE
15C7 96 5A               LDA A  DATFLG    CHECK FCC,FCB,FDB
15C9 26 01               BNE    EXTDAT    IF SO, GO SERVICE
15CB 39         MEM4     RTS              DONE
15CC CE 02 00   EXTDAT   LDX    #BYTSTK
15CF DF 71               STX    XTEMP4    SET BUFFER POINTER
15D1 DE 71      MEM5     LDX    XTEMP4    GET POINTER
15D3 9C 87               CPX    BYTPTR    CHECK EMPTY
15D5 27 F4               BEQ    MEM4      IF SO, DONE
15D7 A6 00               LDA A  0,X
15D9 08                  INX
15DA DF 71               STX    XTEMP4    ADVANCE PTR AND SAVE
15DC DE 8B               LDX    MEMPTR    GET DEST PTR
15DE A7 00               STA A  0,X       PUT BYTE
15E0 08                  INX
15E1 DF 8B               STX    MEMPTR    SAVE PLACE
15E3 8D 02               BSR    INCCNT    FIX THE COUNT
```

```
LOCN B1 B2 B3
15E5 20 EA            BRA     MEM5        DO TILL DONE
15E7 96 9B     INCCNT LDA A   MCOUNT+1
15E9 8B 01            ADD A   #1
15EB 97 9B            STA A   MCOUNT+1
15ED 96 9A            LDA A   MCOUNT
15EF 89 00            ADC A   #0
15F1 97 9A            STA A   MCOUNT      16 BIT INCREMENT
15F3 39               RTS
15F4 DE A2     FIXCNT LDX     LASTM       GET LAST START
15F6 96 9A            LDA A   MCOUNT
15F8 A7 00            STA A   0,X
15FA 96 9B            LDA A   MCOUNT+1
15FC A7 01            STA A   1,X         SET BYTE COUNT
15FE 39               RTS                 DONE
                *
                *
                *
                *
                      END
SYMBOL TABLE:

ADDPC0 0C88   ADDPC1 0C86   ADDPC2 0C7E   ADDPC3 0C72   ADD16  0CFA
ADVPTR 03A6   ASC    13E7   ASMERR 07E5   ASME2  0821   ASME3  0824
ASME4  082B   ASME5  0832   BCONV  12C9   BIN    13BA   BINGO  0962
BIN2   13BC   BUFCNT 00A7   BYTCNT 00A6   BYTPTR 0087   BYTSTK 0200
CERR   047A   CHKCOM 0BAA   CHKERR 0437   CHKFRE 08A4   CHKGEN 14F4
CHKLBL 08DE   CHKLOP 1427   CHKTAP 04ED   CHK1   0939   CHK2   045D
CHK2ER 0433   CHK3   0464   CKDONE 0904   CKSUM  0061   CLRLAB 0C65
CLRLBL 0351   CNXT   0496   CONDON 04BF   CONT   05A5   CONTRL 04B2
COPDON 0CA8   COPLBL 0C8F   CRLF   07CF   DATFLG 005A   DECM   135B
DECM2  1363   DECM3  1387   DECX   04CF   DECXX  1460   DELAY  04C8
DELDON 04D5   DIRECT 0E04   EJCHR  000A   EJECT  1131   EJFLG  005C
EJSTR  11D1   ENDFLG 0058   EQU1   10B9   ERRCNT 00A5   ERRFLG 0056
ERRHD  054B   ERRORS 00A9   ERRPTR 0085   ERRSTK 0100   EVAL   11D5
EVAL1A 11E0   EVAL1B 12A1   EVAL2  12A9   EVAL3  12AD   EVAL4  12BF
EXTDAT 15CC   EXTEND 0D3C   EXTEN0 0D43   EXTEN1 0D49   FCCFLG 005B
FERROR 091A   FILTIT 1114   FIN    04D6   FINDCR 0C26   FINDSC 11E2
FINDSP 0C4F   FINDS2 0C50   FINDS4 11FE   FIN2   0505   FIN3   051E
FIN4   052F   FIN5   0516   FIN6   0537   FIX    0CC1   FIXCNT 15F4
FIXMOD 0D5C   FIXM3  0D67   FIXM4  0D62   FIXM5  0D6D   FIXXX  0E84
FIXXX2 0E01   FNDEND 102D   FNDLBL 0905   FNDOPT 091F   FND10  0908
FND222 0C44   F1     1211   F2     1203   F3     1298   GAP    00AA
GAPX   0531   GENER  00B0   GENOBJ 14FD   GETCHR 0CAC   GETERR 044D
GETER2 046B   GETSYM 057A   GOTLBL 091D   GOTMSG 0664   HASH   0867
HASHCT 00A4   HEADER 1564   HERROR 08B8   HEX    139A   HEXL   0CDB
HEXR   0CDF   HEX2   139C   HEX3   13AA   HEX4   13B9   IMMED  0E3F
IMMED0 0E47   IMMED1 0E4E   IMMED2 0E67   IMMED3 0E6E   IMMED4 0E69
IMMED5 0E76   IMMED6 0E73   INCCNT 15E7   INDEC  1348   INDEC2 1350
INDEX  0DA3   INDEX0 0DF3   INDEX1 0DBA   INDEX2 0DCB   INDEX3 0DD6
INDEX4 0DE0   INDEX5 0DFA   INDEX9 0DF9   INDE00 0DE7   INITR  1388
LABEL  004F   LABERR 0BBC   LABOUT 0584   LASTM  00A2   LASTPC 009E
LBLBEG 0040   LBLEND 0042   LBLMSK 0060   LINBYT 0048   LINCNT 00A8
LINES  0036   LINPTR 008D   LIST   00AE   LNHDX  156F   LOAD   1216
LOAD1  1240   LONE   1393   LOOP   084F   LSTERR 0084   LSTPCM 009C
LSTREC 04F1   LSTSYM 0575   LTEMP  0075   LTWO   1391   L1     1275
```

L2	1278	L3	1283	L4	1287	L5	128B	L6	1290
MAIN	0300	MARDON	114C	MATCH1	0952	MATFLG	0057	MCOUNT	009A
MEMCOD	1577	MEMGEN	0414	MEMOBJ	0049	MEMORY	00B3	MEMPTR	008B
MEM1	1594	MEM2	15A3	MEM3	15C5	MEM4	15CB	MEM5	15D1
MESG0	0687	MESG1	069D	MESG10	0778	MESG11	078E	MESG2	06AE
MESG3	06C6	MESG4	06DE	MESG5	06F9	MESG6	0716	MESG7	072F
MESG8	073C	MESG9	0753	MIX2	0882	MIX3	089F	MODFY	00AB
MON	031B	MOVE	150D	MOVELP	143A	MOVPTR	05AD	MSGHD	0681
MSGTBL	0669	NAM1	10F9	NAM2	1110	NAM3	111E	NDIR	0E38
NOEJT	119B	NOERHD	0549	NOERR	0475	NOERR2	049D	NOERR4	04A4
NOLAB	03A4	NOMATL	0942	NOPRT	05AA	NXTBLK	0C5B	NXTBL2	0C5C
NXTBL3	0C64	OBJADR	00A0	OBJBUF	00B4	OBJCOD	1489	OBJC01	1499
OBJC03	14AA	OBJC04	14A8	OBJC05	14B2	OBJC06	14D2	OBJC07	14DD
OBJDON	1517	OBJGEN	040D	OBJINT	0062	OBJLP	1545	OBJPTR	0089
OCT	13D0	OCT1	13D2	OFFLOP	1471	OPADD	12D3	OPCNT	0090
OPCODE	007E	OPDIV	130F	OPDIV1	1325	OPDIV3	1337	OPMUL	12E7
OPMUL2	12F1	OPMUL3	12FF	OPN	0063	OPNEND	1083	OPNPTR	0096
OPNTBL	12C1	OPSERR	07D6	OPSUB	12DD	OPTABL	096B	OPTDON	1040
OPTEND	0B6F	OPTERR	094D	OPTLST	1041	OPTPTR	0094	OP1	007F
OP2	0080	ORDOK	1466	OUTCH	0320	OUTHEX	0CD0	OUTHL	119D
OUTHR	11A3	OUTHXS	0CCC	OUTS	031E	OUTSZ	0CC9	OUT2S	0CC7
OUT3S	0CC5	PAGENO	00AC	PAGER	00B1	PAGFLG	005F	PARFF2	0C4B
PARSE	0B75	PARSE0	0B7F	PARSE1	0BAE	PARSE2	0BD4	PARSE3	0C2D
PARSE4	0C39	PARSE5	0C38	PARSE6	0C36	PARSE7	0C33	PARS0A	0B77
PARS1A	0BC3	PARS1B	0BD1	PARS2A	0C04	PARS2B	0C10	PARS2D	0C11
PARS2E	0C24	PARS2F	0BFF	PARS2H	0C3C	PARS2J	0C1D	PASDON	1477
PASONE	03B1	PASS	008F	PASS1	03B9	PASS11	03C7	PASS12	03CE
PASS13	03D8	PASS2	03E0	PASS2A	03E8	PASS2B	03FA	PASS2C	041B
PASS2X	0401	PASTHR	05BB	PASTWO	03D9	PC	004B	PCFLAG	0059
PCRLF	07BA	PCRLF1	07C8	PCRLF2	07CC	PCTRL	04B6	PDATA	07AB
PEVAL	0BF2	PLOOP	07A7	PPAG2	1174	PPAG3	1196	PPAG4	1183
PPAG5	117F	PPAG6	116F	PPP	11A9	PRFLG	0055	PRTDAT	05FF
PRTERR	0651	PRTFLG	005E	PRTINA	05CF	PRTINB	05D5	PRTINC	05D7
PRTIND	05CE	PRTINE	05D3	PRTINF	05C1	PRTING	05F8	PRTIT	0CD8
PRTMAR	1146	PRTMES	04EA	PRTPC	0611	PRTREC	1518	PRTSRC	0642
PRTS1	0644	PRTS2	0650	PRT1	0636	PRT2	0639	PRT2ER	048F
PRT3	063C	PRT4	063F	PSTR	07B2	PTNXT	0660	PUTIT	08BB
PUTLBL	08A2	P1INIT	0326	P2DON	04B1	P2ERR1	0081	P2ERR2	0082
P2ERR3	0083	P2INIT	036F	P2IN3	03B0	P3FLG	005D	P3INIT	036F
QTEMP	007B	QTEMP2	0079	QTEMP3	0077	RANDOM	084B	RECORD	151C
REHASH	087F	RNDM	0091	SAME	1481	SAVEPL	1515	SAVPTR	0098
SETBIT	039C	SETGAP	1403	SETTL	035D	SET0	0512	SHELL	13F0
SHELL1	13FC	SHELL2	1401	SHELL5	1458	SHELL6	146D	SHIFTL	150B
SHORT	0422	SORT	141D	SPCL	1351	SPSAVE	0067	SRCBEG	0044
SRCEND	0046	SRCPTR	004D	SRTDON	1480	SUB16	0CE8	SYMBOL	00AF
SYMGEN	055E	SYMHD	0538	SYMPRT	05B8	TAPBYT	1550	TAPE	00B2
TAPEOF	04C4	TAPEON	04C0	TAPOUT	0323	TEMP	007D	TERM	0064
TFIXMD	0D59	TITLE	00C6	TOOMAN	0836	TYPERR	0FEA	TYPE1	0D03
TYPE10	0F7E	TYPE11	0FBD	TYPE12	0FED	TYPE13	1089	TYPE14	10A2
TYPE15	10B0	TYPE16	10DD	TYPE17	10E9	TYPE18	111F	TYPE2	0D06
TYPE2A	0D28	TYPE2B	0D2C	TYPE2D	0D34	TYPE3	0D35	TYPE3A	0D39
TYPE4	0D54	TYPE5	0D51	TYPE6	0D7B	TYPE7	0D88	TYPE7A	0D8D
TYPE7C	0D9C	TYPE7D	0D96	TYPE8	0E87	TYPE8A	0EB1	TYPE8B	0EBE
TYPE8C	0ED6	TYPE8D	0ED4	TYPE8E	0EC4	TYPE8F	0EF1	TYPE8G	0F06
TYPE8H	0F0D	TYPE8I	0F2C	TYPE8J	0F38	TYPE8K	0F34	TYPE8L	0F3A
TYPE9	0F42	TYPE9A	0F62	TYPE9B	0F68	TYPE9C	0F4E	TYPE9D	0F5D
TYP10A	0F8A	TYP10B	0F9E	TYP10C	0F99	TYP11A	0FD9	TYP11B	0FE3
TYP11C	0FE9	TYP12A	1005	TYP12B	1019	TYP12C	1009	TYP12D	0FF8
TYP13A	10A1	TYP15A	10B4	TYP15C	10D7	TYP3R	0D4C	XLOOP	04CC

```
XSAVE  0065   XTEMP  0069   XTEMP1 006B   XTEMP2 006D   XTEMP3 006F
XTEMP4 0071   XTEMP5 0073
```

OBJECT CODE:

```
S1 13 0300 8E A0 7F BD 03 26 BD 03 B1 BD 03 6F BD 03 D9 BD 60
S1 13 0310 03 26 BD 03 B1 BD 03 6F BD 05 BB 7E E0 D0 86 20 BF
S1 13 0320 7E E1 D1 7E E1 D1 86 FF 97 AE 97 B0 97 AF 97 59 22
S1 13 0330 40 97 A8 4F 97 B1 97 AC 97 AD 97 A5 97 56 97 B2 AA
S1 13 0340 97 B3 97 58 97 A9 86 7F 97 60 CE 01 00 DF 85 DE 23
S1 13 0350 40 6F 00 08 9C 42 26 F9 CE 00 C6 86 20 A7 00 08 FC
S1 13 0360 8C 00 E6 26 F8 86 04 A7 00 CE 00 00 DF 4B 39 86 11
S1 13 0370 FF 97 62 97 5D CE 01 00 DF 85 CE 00 00 DF 4B CE 94
S1 13 0380 FF FF DF 9C DF 9E 4F 97 A7 97 9A 97 9B 97 58 CE C6
S1 13 0390 00 B4 DF 89 DE 49 DF 8B DF A2 DE 40 A6 00 27 04 3C
S1 13 03A0 8A 80 A7 00 C6 08 08 9C 42 27 05 5A 26 F8 20 EC 34
S1 13 03B0 39 9F 67 DE 44 09 7F 00 8F DF 4D BD 0B 75 DF 6F 0A
S1 13 03C0 96 4F 27 03 BD 08 A2 96 55 26 03 BD 0C 44 DE 6F 45
S1 13 03D0 96 58 26 04 9C 46 26 E1 39 DE 44 09 86 01 97 8F 07
S1 13 03E0 DF 4D DE 4B DF 6D DE 4D BD 0B 75 DF 6F 96 4F 27 A6
S1 13 03F0 09 BD 09 05 A6 00 84 7F A7 00 96 55 26 03 BD 09 FB
S1 13 0400 1F 96 90 27 16 96 5D 27 04 96 B2 27 07 BD 14 89 78
S1 13 0410 96 5D 27 07 96 B3 27 03 BD 15 77 96 5D 26 03 7E 61
S1 13 0420 04 A4 96 5E 27 0D 96 AE 27 09 96 90 36 BD 05 C1 A5
S1 13 0430 32 97 90 86 FF 97 56 96 A5 27 3A DE 85 EE 00 9C 64
S1 13 0440 4D 26 32 96 AE 26 06 BD 05 FF BD 06 42 DE 85 7A F0
S1 13 0450 00 A5 E6 02 27 15 D1 81 26 03 7F 00 81 D1 82 26 DB
S1 13 0460 03 7F 00 82 D1 83 26 03 7F 00 83 08 08 08 DF 85 89
S1 13 0470 BD 06 51 20 C2 CE 00 81 86 03 36 DF 77 E6 00 27 11
S1 13 0480 15 96 56 27 0A 96 AE 26 06 BD 05 FF BD 06 42 DE 22
S1 13 0490 77 E6 00 BD 06 51 DE 77 08 32 4A 26 DD 96 5F 26 F0
S1 13 04A0 03 BD 11 31 DE 6F 96 58 26 2C 9C 46 27 03 7E 03 2C
S1 13 04B0 E0 39 C6 04 27 09 A6 00 BD 03 23 08 5A 26 F7 39 E4
S1 13 04C0 00 00 00 00 00 00 00 00 C6 04 27 09 CE F4 FF 09 64
S1 13 04D0 26 FD 5A 26 F7 39 96 5D 27 17 BD 07 BA BD 06 39 9A
S1 13 04E0 CE 05 49 96 A9 27 03 CE 05 4B BD 07 AB 96 B2 27 87
S1 13 04F0 14 BD 15 18 86 53 BD 03 23 86 39 BD 03 23 8D C8 47
S1 13 0500 CE 04 C4 8D AD 96 5D 27 2E 96 B3 27 09 BD 15 F4 90
S1 13 0510 DE 8B 6F 00 6F 01 96 AF 26 44 96 AE 27 19 BD 07 98
S1 13 0520 BA 96 B1 27 0A 96 B1 27 06 CE 11 D1 7E 07 AB C6 7B
S1 13 0530 04 BD 07 BA 5A 26 FA 39 20 20 20 53 59 4D 42 4F 98
S1 13 0540 4C 20 54 41 42 4C 45 3A 04 4E 4F 20 45 52 52 4F A0
S1 13 0550 52 28 53 29 20 44 45 54 45 43 54 45 44 04 96 5D 48
S1 13 0560 27 BC C6 04 BD 0F D9 CE 05 38 BD 07 AB BD 13 F0 FB
S1 13 0570 DE 40 09 DF 69 BD 07 BA C6 04 DE 69 08 A6 00 27 A4
S1 13 0580 29 37 C6 06 A6 00 BD 03 20 08 5A 26 F7 BD 0C C7 A6
S1 13 0590 A6 00 BD 0C D0 08 A6 00 BD 0C D0 DF 69 BD 06 39 8D
S1 13 05A0 33 9C 42 27 13 5A 26 D2 20 CB 37 C6 07 08 5A 26 33
S1 13 05B0 FC 33 DF 69 9C 42 26 C2 7E 05 1E 7F 00 5D 7E 03 FC
S1 13 05C0 D9 8D 3C 8D 7D CE 02 00 DF 71 96 5A 26 01 39 96 75
S1 13 05D0 B0 27 FB 96 90 DE 6D 08 4A 26 FC DF 6D 86 01 97 F6
S1 13 05E0 90 DE 71 9C 87 27 E7 A6 00 97 7E 08 9C 87 27 08 E2
S1 13 05F0 7C 00 90 A6 00 97 7F 08 DF 71 BD 05 FF 20 D4 BD 65
S1 13 0600 07 BA BD 03 1E 96 59 26 08 BD 0C C7 BD 0C C5 20 EC
S1 13 0610 25 96 6D BD 0C D0 96 6E BD 0C CC D6 90 27 17 96 42
S1 13 0620 7E BD 0C CC 5A 27 12 96 7F BD 0C CC 5A 27 0D 96 52
S1 13 0630 80 BD 0C CC 20 09 BD 0C C5 BD 0C C5 BD 0C C5 7E 50
```

```
S1 13 0640 03 1E DE 8D A6 00 08 81 0D 27 05 BD 03 20 20 F4 BE
S1 13 0650 39 CE 06 81 BD 07 B2 7F 00 56 CE 06 69 58 27 04 FD
S1 13 0660 08 5A 26 FC EE 00 7E 07 AB 06 87 06 9D 06 AE 06 FA
S1 13 0670 C6 06 DE 06 F9 07 16 07 2F 07 3C 07 53 07 78 07 57
S1 13 0680 8E 2A 2A 20 20 20 04 53 59 4D 42 4F 4C 20 54 41 95
S1 13 0690 42 4C 45 20 4F 56 45 52 46 4C 4F 57 04 55 4E 44 04
S1 13 06A0 45 46 49 4E 45 44 20 53 59 4D 42 4F 4C 04 4D 55 FF
S1 13 06B0 4C 54 49 50 4C 59 20 44 45 46 49 4E 45 44 20 53 D6
S1 13 06C0 59 4D 42 4F 4C 04 55 4E 52 45 43 4F 47 4E 49 5A 9B
S1 13 06D0 41 42 4C 45 20 4D 4E 45 4D 4F 4E 49 43 04 49 4C F3
S1 13 06E0 4C 45 47 41 4C 20 43 48 41 52 41 43 54 45 52 20 D4
S1 13 06F0 49 4E 20 4C 41 42 45 4C 04 49 4C 4C 45 47 41 4C E1
S1 13 0700 20 43 48 41 52 41 43 54 45 52 20 49 4E 20 4F 50 C2
S1 13 0710 45 52 41 4E 44 04 52 45 4C 41 54 49 56 45 20 42 A9
S1 13 0720 52 41 4E 43 48 20 54 4F 4F 20 4C 4F 4E 47 04 53 A0
S1 13 0730 59 4E 54 41 58 20 45 52 52 4F 52 04 49 4C 4C 45 4D
S1 13 0740 47 41 4C 20 49 4E 44 45 58 20 56 41 52 49 41 42 64
S1 13 0750 4C 45 04 49 4C 4C 45 47 41 4C 20 43 48 41 52 41 87
S1 13 0760 43 54 45 52 20 46 4F 52 20 53 50 45 43 49 46 49 2D
S1 13 0770 45 44 20 42 41 53 45 04 49 4C 4C 45 47 41 4C 20 93
S1 13 0780 4F 50 54 49 4F 4E 20 53 57 49 54 43 48 04 54 4F F3
S1 13 0790 4F 20 4D 41 4E 59 20 4F 50 45 52 41 4E 44 53 20 15
S1 13 07A0 28 44 41 54 41 29 04 BD 03 20 08 A6 00 81 04 26 9D
S1 13 07B0 F6 39 DF 65 8D 04 DE 65 20 F1 CE 07 CF 8D EC 96 2A
S1 13 07C0 A8 4C 97 A8 81 36 22 04 7F 00 5C 39 7E 11 31 0D 34
S1 13 07D0 0A 00 00 00 00 04 36 86 01 97 7E 97 7F 97 80 97 71
S1 13 07E0 59 BD 0C 72 32 36 97 84 32 7D 00 56 26 33 C6 FF CB
S1 13 07F0 D7 A9 7D 00 8F 26 2D D6 A5 C1 55 27 24 36 96 4D 21
S1 13 0800 D6 4E DE 85 A7 00 E7 01 32 A7 02 08 08 08 DF 85 77
S1 13 0810 96 A5 4C 97 A5 81 55 26 08 CE 08 36 8D 94 9E 67 DB
S1 13 0820 39 86 FF 39 D6 81 26 03 97 81 39 D6 82 26 03 97 E4
S1 13 0830 82 39 97 83 39 39 45 52 52 4F 52 20 4C 49 4D 49 98
S1 13 0840 54 20 45 58 43 45 45 44 45 44 04 37 36 C6 18 96 14
S1 13 0850 91 48 48 48 98 91 48 48 79 00 93 79 00 92 79 00 E2
S1 13 0860 91 5A 26 EB 32 33 39 CE 00 4F 7F 00 A4 A6 00 AB 59
S1 13 0870 05 97 93 A6 01 A9 04 97 92 A6 02 A9 03 97 91 7C D0
S1 13 0880 00 A4 BD 08 4B 96 93 84 F8 D6 92 C4 1F 9B 41 D9 0B
S1 13 0890 40 97 6A D7 69 D1 42 22 E9 25 04 91 43 22 E3 DE D5
S1 13 08A0 69 39 8D C3 A6 00 27 13 BD 08 DE 27 0B BD 08 7F 59
S1 13 08B0 96 A4 81 28 26 EE 86 00 7E 07 E5 96 4F A7 00 96 2B
S1 13 08C0 50 A7 01 96 51 A7 02 96 52 A7 03 96 53 A7 04 96 E0
S1 13 08D0 54 A7 05 96 4B A7 06 96 4C A7 07 DF 75 39 86 02 E1
S1 13 08E0 E6 00 D4 60 D1 4F 26 1C D6 50 E1 01 26 16 D6 51 1D
S1 13 08F0 E1 02 26 10 D6 52 E1 03 26 0A D6 53 E1 04 26 04 67
S1 13 0900 D6 54 E1 05 39 BD 08 67 A6 00 27 0E BD 08 DE 27 C9
S1 13 0910 0C BD 08 7F 96 A4 81 28 26 EE 86 FF 39 4F 39 4F F7
S1 13 0920 97 5A 97 57 97 5B 97 5C DE 96 DF 6B DE 94 A6 02 27
S1 13 0930 97 7D E6 01 A6 00 CE 09 6B A1 00 27 15 7D 00 57 1F
S1 13 0940 26 0B 08 08 08 08 08 08 8C 0B 75 26 EC 86 03 7E 1D
S1 13 0950 07 D6 97 57 E1 01 26 EA 36 96 7D A1 02 27 03 32 8E
S1 13 0960 20 E0 32 A6 03 97 7E EE 04 6E 00 41 42 41 1B 0D 47
S1 13 0970 03 41 44 43 89 0D 51 41 44 44 8B 0D 51 41 4E 44 3C
S1 13 0980 84 0D 51 41 53 4C 48 0D 7B 41 53 52 47 0D 7B 42 DA
S1 13 0990 43 43 24 0D 06 42 43 53 25 0D 06 42 45 51 27 0D 7A
S1 13 09A0 06 42 47 45 2C 0D 06 42 47 54 2E 0D 06 42 48 49 3F
S1 13 09B0 22 0D 06 42 48 53 24 0D 06 42 49 54 85 0D 51 42 E6
S1 13 09C0 4C 45 2F 0D 06 42 4C 4F 25 0D 06 42 4C 53 23 0D 2A
S1 13 09D0 06 42 4C 54 2D 0D 06 42 4D 49 2B 0D 06 42 4E 45 00
S1 13 09E0 26 0D 06 42 50 4C 2A 0D 06 42 52 41 20 0D 06 42 65
```

```
S1 13 09F0 53 52 8D 0D 06 42 56 43 28 0D 06 42 56 53 29 0D 77
S1 13 0A00 06 43 42 41 11 0D 03 43 4C 43 0C 0D 03 43 4C 49 2F
S1 13 0A10 0E 0D 03 43 4C 52 4F 0D 7B 43 4C 56 0A 0D 03 43 BA
S1 13 0A20 4D 50 81 0D 51 43 4F 4D 43 0D 7B 43 50 58 8C 0D 18
S1 13 0A30 51 44 41 41 19 0D 03 44 45 43 4A 0D 7B 44 45 53 F8
S1 13 0A40 34 0D 03 44 45 58 09 0D 03 45 4E 44 00 10 DD 45 5B
S1 13 0A50 4F 52 88 0D 51 45 51 55 00 10 B0 46 43 42 00 0F 86
S1 13 0A60 42 46 43 43 00 0E 87 46 44 42 00 0F 7E 49 4E 43 AC
S1 13 0A70 4C 0D 7B 49 4E 53 31 0D 03 49 4E 58 08 0D 03 4A 22
S1 13 0A80 4D 50 6E 0D 35 4A 53 52 AD 0D 35 4C 44 41 86 0D D3
S1 13 0A90 51 4C 44 53 8E 0D 51 4C 44 58 CE 0D 51 4C 53 52 2D
S1 13 0AA0 44 0D 7B 4D 4F 4E 00 10 DD 4E 41 4D 00 10 E9 4E 7C
S1 13 0AB0 45 47 40 0D 7B 4E 4F 50 01 0D 03 4F 50 54 00 0F DE
S1 13 0AC0 ED 4F 52 41 8A 0D 51 4F 52 47 00 10 A2 50 41 47 F9
S1 13 0AD0 00 10 89 50 53 48 36 0D 88 50 55 4C 32 0D 88 52 B9
S1 13 0AE0 4D 42 00 11 1F 52 4F 4C 49 0D 7B 52 4F 52 46 0D 3F
S1 13 0AF0 7B 52 54 49 3B 0D 03 52 54 53 39 0D 03 53 42 41 25
S1 13 0B00 10 0D 03 53 42 43 82 0D 51 53 45 43 0D 0D 03 53 BE
S1 13 0B10 45 49 0F 0D 03 53 45 56 0B 0D 03 53 50 43 00 0F 26
S1 13 0B20 BD 53 54 41 97 0D 54 53 54 53 9F 0D 54 53 54 58 2B
S1 13 0B30 DF 0D 54 53 55 42 80 0D 51 53 57 49 3F 0D 03 54 13
S1 13 0B40 41 42 16 0D 03 54 41 50 06 0D 03 54 42 41 17 0D 02
S1 13 0B50 03 54 50 41 07 0D 03 54 53 54 4D 0D 7B 54 53 58 C3
S1 13 0B60 30 0D 03 54 54 4C 00 10 E9 54 58 53 35 0D 03 57 B9
S1 13 0B70 41 49 3E 0D 03 96 48 08 4A 2A FC DF 7B DF 8D 86 F7
S1 13 0B80 FF 97 55 97 5E 97 5F BD 0C 65 4F 97 90 97 AB 97 0E
S1 13 0B90 7D 97 59 97 81 97 82 97 83 97 56 DF 94 DF 96 DE E6
S1 13 0BA0 7B A6 00 81 0D 26 03 7E 0C 2D 81 2A 27 78 81 20 C7
S1 13 0BB0 27 22 97 59 81 41 25 04 81 5A 23 07 86 04 BD 07 BA
S1 13 0BC0 E5 20 0E BD 0C 8F 4D 26 08 C1 0D 27 60 C1 20 26 DF
S1 13 0BD0 EB BD 0C 50 BD 0C 5C 27 54 5F D7 55 86 FF 97 59 6D
S1 13 0BE0 DF 94 08 A6 00 81 0D 27 16 08 A6 00 81 0D 27 0F A3
S1 13 0BF0 20 12 96 8F 4A 97 56 BD 11 D5 7F 00 56 39 02 86 2A
S1 13 0C00 03 20 48 02 8D 55 27 25 81 41 27 05 81 42 26 14 5A
S1 13 0C10 5C 5C 08 A6 00 81 0D 27 20 81 20 27 1F 09 20 04 81
S1 13 0C24 DF 96 08 A6 00 81 0D 26 F9 96 7D 27 07 DF 7B BD 94
S1 13 0C34 07 D6 DE 7B 39 D7 AB 39 D7 AB 8D 1C 27 EB 20 E0 45
S1 13 0C44 DE 4B DF 6D 7E 09 1F 97 7D 20 D7 08 A6 00 81 0D 3A
S1 13 0C54 27 0E 81 20 26 F5 39 08 A6 00 81 20 27 F9 81 0D 65
S1 13 0C64 39 CE 00 20 DF 4F CE 20 20 DF 51 DF 53 39 DE 4B 55
S1 13 0C74 08 08 7C 00 90 7C 00 90 20 0A DE 4B 08 7C 00 90 DD
S1 13 0C84 20 02 DE 4B 08 DF 4B 7C 00 90 39 8D 1B 97 4F 8D 7F
S1 13 0C94 17 97 50 8D 13 97 51 8D 0F 97 52 8D 0B 97 53 8D 32
S1 09 0CA4 07 97 54 39 08 39 DA
S1 13 0CAC A6 00 84 7F 16 81 30 25 0C 81 39 23 EF 81 41 25 E0
S1 13 0CBC 04 81 5A 23 E7 31 31 4F 39 8D 02 8D 00 7E 03 1E 96
S1 13 0CCC 8D 02 20 F9 36 8D 08 8D 03 32 8D 07 7E 03 20 44 66
S1 13 0CDC 44 44 44 84 0F 8B 90 19 89 40 19 39 97 7D A6 01 9B
S1 13 0CEC 10 A7 01 A6 00 92 7D A7 00 49 88 01 46 39 EB 01 A3
S1 13 0CFC A9 00 A7 00 E7 01 39 7E 0C 86 96 AB 26 42 BD 0C F1
S1 13 0D0C 7E 96 8F 27 23 BD 11 D5 26 16 96 4B D6 4C CE 00 36
S1 13 0D1C 7B BD 0C E8 4F D6 7C D7 7F 2A 01 43 91 7B 27 08 F7
S1 13 0D2C 7F 00 7F 86 06 7E 07 E5 39 96 AB 26 13 BD 0D A3 9F
S1 13 0D3C 96 8F 27 09 BD 11 D5 DE 7B DF 7F 8D 13 7E 0C 72 58
S1 13 0D4C 86 03 7E 07 D6 BD 0E 3F BD 0E 04 20 E0 8D 01 39 0F
S1 13 0D5C D6 7E C1 80 24 05 96 AB 26 36 39 C4 0F C1 0B 22 2E
S1 13 0D6C F5 96 AB 27 2B 4A 40 84 40 9B 7E 97 7E 4F 39 96 51
S1 13 0D7C AB 4A 2A 0D D6 7E CB 20 D7 7E 20 B1 96 AB 4A 2B 1C
S1 13 0D8C BF D6 7E C1 3F 23 03 40 84 10 1B 97 7E 7E 0C 86 06
```

```
S1 13 0D9C 31 31 86 03 7E 07 D6 DE 6B 7F 00 7F A6 00 81 58 37
S1 13 0DAC 26 0C A6 01 81 20 27 22 81 0D 26 02 20 1C A6 00 D8
S1 13 0DBC 81 2C 27 20 81 20 27 2F 81 0D 27 2B 08 20 EF 96 AB
S1 13 0DCC 8F 27 07 BD 11 D5 96 7C 97 7F BD 0D 59 26 26 31 EB
S1 13 0DDC 31 7E 0C 7E A6 01 81 58 26 14 08 A6 01 81 20 27 99
S1 13 0DEC DE 81 0D 27 DA 20 07 D6 7E CB 10 D7 7E 39 86 08 14
S1 13 0DFC 31 31 7E 07 D6 31 31 39 DE 6B 86 FF 97 56 97 60 D9
S1 13 0E0C DF 73 BD 11 D5 7F 00 56 C6 7F D7 60 DE 6B E6 00 5D
S1 13 0E1C C1 2C 36 07 DE 73 DF 6B 33 06 27 10 5D 26 0D D6 27
S1 13 0E2C 7B 26 09 BD 0D 59 26 50 96 7C 20 2F D6 7E CB 10 DF
S1 13 0E3C D7 7E 39 DE 6B A6 00 81 23 27 07 D6 7E CB 10 D7 4D
S1 13 0E4C 7E 39 08 DF 6B D6 7E C4 0F C1 0B 22 15 BD 0D 59 3C
S1 13 0E5C 26 26 96 8F 27 07 BD 11 D5 96 7C 97 7F BD 0C 7E D1
S1 13 0E6C 20 16 96 AB 4A 2B 03 7E 0D 9C BD 0C 72 96 8F 27 D5
S1 13 0E7C 07 BD 11 D5 DE 7B DF 7F 31 31 39 86 FF 97 56 DE 16
S1 13 0E8C 96 DF 73 BD 11 D5 CE 02 00 DF 87 96 7C 27 56 DE 24
S1 13 0E9C 6B A6 00 81 2C 26 4E 08 96 7C E6 00 08 C1 0D 26 14
S1 13 0EAC 04 97 5B C6 20 D7 7E DF 71 BD 0C 86 DE 71 4A 26 A3
S1 13 0EBC 01 39 97 5A 86 01 97 A6 E6 00 08 DF 71 7D 00 5B 1D
S1 13 0ECC 26 06 C1 0D 26 04 97 5B C6 20 DE 87 E7 00 08 DF E3
S1 13 0EDC 87 BD 0C 86 DE 71 7A 00 5A 26 DD 86 01 97 90 97 C1
S1 13 0EEC 5A 7F 00 56 39 DE 73 E6 00 08 A6 00 97 7E DF 71 40
S1 13 0EFC BD 0C 86 DE 71 E1 01 26 01 39 D7 5A 86 01 97 A6 0D
S1 13 0F0C 08 A6 00 08 DF 71 DE 87 11 27 15 81 0D 27 11 A7 AC
S1 13 0F1C 00 08 DF 87 8C 03 00 27 13 BD 0C 86 DE 71 20 E1 EB
S1 13 0F2C 7F 00 56 86 01 97 90 39 8D 63 20 02 8D F2 7F 00 E5
S1 13 0F3C 56 86 0B 7E 07 E5 CE 02 00 DF 87 BD 0B F2 96 7C 4E
S1 13 0F4C 97 7E BD 0C 86 DE 6B A6 00 81 0D 27 04 81 2C 27 B1
S1 13 0F5C 05 86 01 97 90 39 97 5A 86 01 97 A6 08 DF 6B BD D1
S1 13 0F6C 0B F2 DE 87 96 7C A7 00 08 DF 87 8C 03 00 27 BC 76
S1 13 0F7C 20 D0 CE 02 00 DF 87 BD 0B F2 DE 7B DF 7E BD 0C 02
S1 13 0F8C 7E DE 6B A6 00 81 0D 27 04 81 2C 27 05 86 02 97 33
S1 13 0F9C 90 39 97 5A 86 02 97 A6 08 DF 6B BD 0B F2 DE 87 51
S1 13 0FAC 96 7B A7 00 96 7C A7 01 08 08 DF 87 8C 03 00 20 9A
S1 13 0FBC CD 7F 00 59 96 8F 27 25 96 5D 27 21 96 4F 26 1E A7
S1 13 0FCC 96 AE 27 19 BD 11 D5 D6 7C 26 02 C6 01 BD 07 BA 2B
S1 13 0FDC 96 5C 26 03 5A 26 F6 7F 00 5C 7F 00 5E 39 7E 10 F1
S1 13 0FEC B4 7F 00 59 96 8F 26 F5 96 4F 26 F2 DE 6B A6 02 37
S1 13 0FFC 97 7D A6 00 E6 01 CE 10 41 A1 00 27 10 08 08 08 31
S1 13 100C 08 08 08 8C 10 89 26 F1 86 0A 7E 07 E5 E1 01 26 7A
S1 13 101C EC 36 96 7D A1 02 32 26 E4 A6 03 EE 04 A7 00 DE 8C
S1 13 102C 6B A6 00 08 DF 6B 81 0D 27 0A 81 20 27 06 81 2C 13
S1 13 103C 27 BA 20 ED 39 4C 49 53 FF 00 AE 4E 4F 4C 00 00 FB
S1 13 104C AE 54 41 50 FF 00 B2 4E 4F 54 00 00 B2 4D 45 4D CA
S1 13 105C FF 00 B3 4E 4F 4D 00 00 B3 53 59 4D FF 00 AF 4E 3C
S1 13 106C 4F 53 00 00 AF 47 45 4E FF 00 B0 4E 4F 47 00 00 B2
S1 13 107C B0 50 41 47 FF 00 B1 4E 4F 50 00 00 B1 7F 00 59 B2
S1 13 108C 96 8F 27 11 96 4F 26 20 97 5E 96 B1 27 07 96 AE 1A
S1 13 109C 27 03 7F 00 5F 39 96 4F 26 0E BD 11 D5 DE 7B DF 0B
S1 13 10AC 4B DF 6D 39 96 4F 26 05 86 07 7E 07 E5 BD 09 05 8E
S1 13 10BC DF FD 96 8F 4A 97 56 BD 11 D5 7F 00 56 DE FD 96 FF
S1 13 10CC 7C D6 7B E7 06 A7 07 DE 7B DF 6D 39 96 84 7E 07 2B
S1 13 10DC E5 7F 00 59 96 4F 26 D0 86 FF 97 58 39 7F 00 59 E3
S1 13 10EC 96 8F 27 2E 96 4F 26 C0 CE 00 C6 DF 65 DE 96 A6 B9
S1 13 10FC 00 81 0D 27 0F 08 DF 96 DE 65 A7 00 08 DF 65 8C DD
S1 13 110C 00 E6 26 E9 86 20 DE 65 8C 00 E6 27 05 A7 00 08 A4
S1 13 111C 20 F6 39 BD 11 D5 CE 00 7B D6 4C 96 4B BD 0C FA BE
S1 13 112C DE 7B DF 4B 39 37 D6 B1 27 65 CE 11 D1 BD 07 AB 8A
S1 13 113C 37 4F 97 A8 97 B1 C6 03 27 06 BD 07 BA 5A 26 FA A4
```

```
S1 13 114C CE 00 C6 BD 07 AB CE 11 A9 BD 07 AB 96 AD 8B 01 C6
S1 13 115C 19 97 AD 96 AC 89 00 19 97 AC 27 0C 84 F0 27 03 2A
S1 13 116C 8D 2F 5C 96 AC 8D 30 5C 96 AD 27 1E 5D 26 04 85 68
S1 13 117C F0 27 04 8D 1C 96 AD 8D 1E BD 07 BA BD 07 BA 86 2B
S1 13 118C FF 97 5C 97 5F 33 D7 B1 33 39 5D 26 E6 20 E8 33 9C
S1 13 119C 39 BD 0C DB 7E 03 20 BD 0C DF 7E 03 20 20 20 20 18
S1 13 11AC 20 20 20 20 20 54 53 43 20 4D 4E 45 4D 4F 4E 49 72
S1 13 11BC 43 20 41 53 53 45 4D 42 4C 45 52 20 20 20 20 50 4E
S1 13 11CC 41 47 45 20 04 00 00 0A 04 4F 97 7B 97 7C 97 63 A2
S1 13 11DC DE 6B DF 96 DE 96 A6 00 08 5F 81 2B 27 27 5C 81 E9
S1 13 11EC 2D 27 22 5C 81 2A 26 0A 09 9C 96 07 08 06 27 E6 E5
S1 13 11FC 20 13 5C 81 2F 27 0E C6 FF 81 20 27 08 81 2C 27 02
S1 13 120C 04 81 0D 26 D1 D7 64 09 DF 6B DE 96 7F 00 7D A6 A1
S1 13 121C 00 81 41 25 1F 81 5A 22 1B DF 79 BD 0C 65 DE 79 C3
S1 13 122C BD 0C 8F BD 09 05 EE 06 DF 79 DE 6B 4D 2A 50 86 A9
S1 13 123C 01 7E 12 98 C6 01 81 24 27 2F 5C 81 25 27 2A 5C 04
S1 13 124C 81 40 27 25 5C 81 27 27 20 DE 6B 09 7C 00 7D 5A 91
S1 13 125C A6 00 81 4F 27 16 81 51 27 12 5A 81 42 27 0D 5A 15
S1 13 126C 81 48 27 08 5A D7 7D 20 03 08 DF 96 4F 97 79 97 32
S1 13 127C 7A CE 12 C9 58 27 04 08 5A 26 FC EE 00 AD 00 96 03
S1 13 128C 7D 27 01 08 DF 71 9C 6B 27 0B 86 09 7F 00 7B 7F 10
S1 13 129C 00 7C 7E 07 E5 96 63 CE 12 C1 48 27 04 08 4A 26 D3
S1 13 12AC FC EE 00 AD 00 DE 6B 08 DF 96 96 64 97 63 2B 03 AF
S1 13 12BC 7E 11 E0 4F 39 12 D3 12 DD 12 E7 13 0F 13 5B 13 B7
S1 13 12CC 9A 13 BA 13 D0 13 E7 96 79 D6 7A CE 00 7B 7E 0C 98
S1 13 12DC FA 96 79 D6 7A CE 00 7B 7E 0C E8 CE 00 00 DF 77 C6
S1 13 12EC CE 00 77 C6 10 A6 03 46 24 09 37 A6 04 E6 05 BD 2E
S1 13 12FC 0C FA 33 64 00 66 01 66 02 66 03 5A 26 E7 EE 02 B2
S1 13 130C DF 7B 39 CE 00 00 DF 77 DE 79 D6 7C D7 7A D6 7B CB
S1 13 131C D7 79 DF 7B C6 11 CE 00 77 37 96 7B D6 7C BD 0C 94
S1 13 132C E8 25 08 96 7B D6 7C BD 0C FA 0C 69 03 69 02 69 26
S1 13 133C 01 69 00 33 5A 26 E2 EE 02 DF 7B 39 E6 00 C0 3A 3B
S1 13 134C 24 02 CB 0A 39 96 6D 97 79 96 6E 97 7A 08 39 8D 63
S1 13 135C 2B A6 00 81 2A 27 EE 8D E3 24 20 37 96 79 D6 7A A2
S1 13 136C 8D 25 8D 23 DB 7A D7 7A 99 79 97 79 8D 19 33 4F 1B
S1 13 137C DB 7A 99 79 D7 7A 97 79 08 20 DC 39 DE 96 7F 00 65
S1 13 138C 79 7F 00 7A 39 8D 00 78 00 7A 79 00 79 39 8D EC 7F
S1 13 139C A6 00 80 47 2A 17 8B 06 2A 04 8B 07 2A 0F 8B 0A 70
S1 13 13AC 2B 0B 8D E1 8D DF 9B 7A 97 7A 08 20 E3 39 8D CC 5A
S1 13 13BC A6 00 80 30 2B F7 81 01 22 F3 46 79 00 7A 79 00 5C
S1 13 13CC 79 08 20 EC 8D B6 A6 00 80 30 2B E1 81 07 22 DD 54
S1 13 13DC 8D B3 8D B3 9B 7A 97 7A 08 20 EB 8D 9F A6 00 97 DB
S1 13 13EC 7A DE 6B 39 7F 00 7D 86 08 36 86 20 36 86 68 36 31
S1 13 13FC 32 97 AA DE 40 DF 77 DF 7B 96 7C 9B AA 97 7A 96 9E
S1 13 140C 7B 89 00 97 79 91 42 25 08 26 60 96 7A 91 43 24 2A
S1 13 141C 5A C6 06 DE 7B DF 69 DE 79 DF 6D DE 69 A6 00 08 5D
S1 13 142C DF 69 DE 6D A1 00 27 4D 23 30 C6 08 DE 7B DF 69 42
S1 13 143C 37 A6 00 DE 79 E6 00 A7 00 08 DF 79 DE 69 E7 00 4D
S1 13 144C 08 33 5A 26 E9 96 7D 26 03 73 00 7D DE 7B 9C 40 87
S1 13 145C 27 08 C6 08 09 5A 26 FC 20 9D 96 7D 27 03 7F 00 81
S1 13 146C 7D DE 77 C6 08 08 5A 26 FC 20 8A 96 AA 81 08 27 AE
S1 13 147C 03 7E 13 FC 39 08 DF 6D 5A 26 A0 20 DD 96 62 27 03
S1 13 148C 0C CE 04 C0 BD 04 B2 BD 04 C8 7F 00 62 DE 6D 9C EA
S1 13 149C 9E 07 DE 4B DF 9E 06 27 03 BD 15 18 96 90 D6 A7 34
S1 13 14AC 26 04 DE 6D DF A0 DE 89 D6 7E E7 00 08 7C 00 A7 6B
S1 13 14BC 4A 27 13 D6 7F E7 00 08 7C 00 A7 4A 27 08 D6 80 62
S1 13 14CC E7 00 08 7C 00 A7 8D 20 96 5A 27 3F CE 02 00 DF 48
S1 13 14DC 71 DE 71 9C 87 27 34 A6 00 08 DF 71 DE 89 A7 00 B2
S1 13 14EC 08 7C 00 A7 8D 02 20 E9 DF 89 96 A7 81 0F 22 01 D1
```

```
S1 13 14FC 39 36 86 10 BD 15 1C 32 CE 00 B4 80 10 97 A7 27 40
S1 13 150C 08 E6 10 E7 00 08 4A 26 F8 DF 89 39 96 A7 27 FB 76
S1 13 151C 36 7F 00 A7 CE 00 B4 DF 89 8D 3D 32 36 8B 03 8D 28
S1 13 152C 23 96 A0 BD 15 50 96 A1 8D 1A 32 36 9B A1 97 A1 76
S1 13 153C 96 A0 89 00 97 A0 33 DE 89 A6 00 8D 07 08 5A 26 49
S1 13 154C F8 96 61 43 36 9B 61 97 61 32 36 BD 0C DB BD 03 63
S1 13 155C 23 32 BD 0C DF 7E 03 23 CE 15 6F C6 08 7F 00 61 DA
S1 13 156C 7E 04 B6 0D 0A 00 00 00 00 53 31 DE 6D 9C 9C 07 0E
S1 13 157C DE 4B DF 9C 06 27 20 DE 8B 96 6D A7 02 96 6E A7 AA
S1 13 158C 03 9C 49 27 03 BD 15 F4 DE 8B DF A2 08 08 08 08 69
S1 13 159C 4F 97 9A 97 9B DF 8B DE 8B D6 90 96 7E A7 00 08 8D
S1 13 15AC BD 15 E7 5A 27 13 96 7F A7 00 08 BD 15 E7 5A 27 E0
S1 13 15BC 08 96 80 A7 00 08 BD 15 E7 DF 8B 96 5A 26 01 39 DB
S1 13 15CC CE 02 00 DF 71 DE 71 9C 87 27 F4 A6 00 08 DF 71 60
S1 13 15DC DE 8B A7 00 08 DF 8B 8D 02 20 EA 96 9B 8B 01 97 8C
S1 13 15EC 9B 96 9A 89 00 97 9A 39 DE A2 96 9A A7 00 96 9B A5
S1 06 15FC A7 01 39 07
S9
```

The TSC Mnemonic Assembler for the FLEX Operating System is a very fast and powerful assembler for the 6800 computer system. The disk version will allow the assembly of very large source files and will produce machine readable binary directly to a disk file. Several assembly time options are also available. This manual is only a supplement to the full Mnemonic Assembler manual and only includes the differences between the FLEX version and the RAM based version.

DESCRIPTION

The general syntax of the ASMB utility is:

ASMB,<file spec 1>[,<file spec 2>][,+<option list>]

The name of the file to be assembled is <file spec 1> and defaults to a TXT extension and to the working drive. If <file spec 2> is specified, that will be the name of the binary file written to the disk. If <file spec 2> is not included in the command line, the binary file will have the same 'name' as that of the file you are assembling, but the extension will be BIN. If a binary file of the same name already exists on the disk, the assembler will ask if it should be deleted. Respond with 'Y' if it should be.

There are 6 options which may appear in the <option list>. The option list is separated from the command line by a plus sign ('+'), and each option is represented by a single letter. Following is a list of the available options and what they mean:

B Do not create a binary file on the disk. If this option appears in the list, no binary file will be created, even if <file spec 2> is specified. This is useful while running the assembler and checking for errors, before the final program is completed.

G Turn 'NOGEN' on. Refer to the full assembler manual for a description of NOGEN.

L Turn the assembler LIST mode off. The assembler will normally produce an assembled listing. The L option will suppres this listing, and only error lines (if any exist) will be displayed.

S Suppress the output of the Symbol Table. Normally the assembler will output a sorted symbol table at the end of each assembly. Including the S in the option list will suppress the output of this table.

N Output line Numbers with the assembled listing. The assembler will normally not include line numbers in the listing. During program debugging it is often useful to have each statement numbered. These numbers will directly correspond to the numbers assigned to each line while in the Text Editing System. Error lines will always be output with line numbers, regardless of the status of the N option.

T Assemble the file for an 'S1' formatted object tape. This option allows the production of an 'S1' tape directly from the assembler. Consult the full manual for Tape details.

Y This option tells the assembler to automatically delete any old binary files of the same name as the source file if binary is being produced. If this option is not specified, the assembler will prompt you if a binary file already exists.

Some examples of assembler command lines follow:

```
ASMB,TEST
ASMB,TEST,+LS
ASMB,0.TEST,1.TEST.CMD,+S
ASMB,TEST,+BN
```

The first example would assemble the source file TEST.TXT from the working drive and create a binary file named TEST.BIN on the same drive. The second example would do the same operation as the first example, but this time the Listing and the Symbol table output would be suppressed. The next example would assemble the file named TEST.TXT on drive 0 and produce a binary file on drive 1 named TEST.CMD. Because of the S in the options field, no symbol table would be output. The last example will assemble the file named TEST.TXT, producing a listing with line numbers, and not produce a binary file (because of the B).

ADDITIONAL ASSEMBLER FEATURES

There is one additional feature included in the FLEX version of the assembler which is not included in the RAM version. In order for a binary file to be directly executable by FLEX, it must have a transfer address associated with it. This assembler allows for the assignment of a transfer address by putting a label or value in the 'operand field' of the END statement in the source listing. As an example, suppose the program you were assembling was to start executing at location $100, and in the source you had the label START on the statement which was ORGed at $100. The END statement should now include this label in its operand field in order to assign $100 as the transfer address.

```
          ORG     $100
START     LDX     $FFFF

          rest of program here

          END     START
```

The above shows what this would look like.

ASSEMBLED LISTINGS TO A PRINTER

To have the listing printed on a printer, as opposed to the terminal, simply precede the command with P (see the 'P' command in the FLEX UCS description). This is assuming that the appropriate PRINT.SYS file exists in the operating system). As an example:

```
+++P,ASMB,TEST
```

This would cause the assembled listing of the source file TEST.TXT to be printed on the printer.

SYMBOL TABLE SPACE

The number of symbols which the assembler will support is dependent upon the amount of memory installed in the computer. As a rough guide, 12K will allow approximately 200 symbols while 24K and above will allow 1000 symbols. It is roughly proportionate for values between these extremes. The assembler automatically determines the memory size.

```
                    *
                    * TSC ASSEMBLER MODS FOR USE WITH THE
                    * FLEX OPERATING SYSTEM.
                    *
                    * COPYRIGHT (C) 1978 BY
                    *
                    * TECHNICAL SYSTEMS CONSULTANTS, INC.
                    * P. O. BOX 2574
                    * WEST LAFAYETTE, INDIANA  47906
                    * 317-423 5465


                    * EQUATES TO DOS

0326                P1INIT  EQU     $0326
03B1                PASONE  EQU     $3B1
036F                P2INIT  EQU     $36F
03D9                PASTWO  EQU     $3D9
0073                XTEMP5  EQU     $73
0096                OPNPTR  EQU     $96
1348                INDEC   EQU     $1348
0EF1                TYPE8F  EQU     $0EF1
05FF                PRTDAT  EQU     $5FF
0CC7                OUT2S   EQU     $CC7
7806                FMS     EQU     $7806
7803                FMSCLS  EQU     $7803
0003                FEER    EQU     3
0004                NFER    EQU     4
7094                BUFPNT  EQU     $7094
708C                ASN     EQU     $708C
0020                SPC     EQU     $20
7740                FCB     EQU     $7740
7091                LSTTRM  EQU     $7091
7082                DEOL    EQU     $7082
7103                WARMS   EQU     $7103
710F                GETCHR  EQU     $710F
7112                PUTCHR  EQU     $7112
7115                INBUF   EQU     $7115
7118                PSTRNG  EQU     $7118
711E                PCRLF   EQU     $711E
7121                NXTCH   EQU     $7121
7127                GETFIL  EQU     $7127
712D                SETEXT  EQU     $712D
7133                OUTDEC  EQU     $7133
7139                OUTHEX  EQU     $7139
713C                RPTERR  EQU     $713C
713F                GETHEX  EQU     $713F
70A3                OUTCHN  EQU     $70A3
7096                RETRNR  EQU     $7096

                    * TEMPORARY STORAGE
```

```
0028                    ORG     $28

0028            OBJFLG  RMB     1
0029            PRNFLG  RMB     1
002A            NLFLG   RMB     1
002B            NSFLG   RMB     1
002C            NGFLG   RMB     1
002D            TPFLG   RMB     1
002E            END     RMB     1
002F            FLG     RMB     1
0030            YFLG    RMB     1
0031            FCBFLG  RMB     1
0032            SPCFLG  RMB     1
0033            LINFLG  RMB     1
0034            LINCNT  RMB     2
0036            XFRFLG  RMB     1
0037            XFRADR  RMB     2
0039            TMPOUT  RMB     2
003B            BUFEND  RMB     2
003D            INDEX5  RMB     2


                * NEW ASSEMBLER STARTS HERE

0300                    ORG     $300
0300 7E 16 02   START   JMP     DASMB

031B                    ORG     $31B
031B 7E 71 03           JMP     WARMS

0320                    ORG     $320
0320 7E 71 12           JMP     PUTCHR

03E2                    ORG     $3E2
03E2 BD 18 71           JSR     P2AUX      SET LINCNT
03E5 01                 NOP

050D                    ORG     $50D
050D BD 19 2D           JSR     FINOBJ
0510 20 04              BRA     *+6

07BA                    ORG     $7BA
07BA BD 71 1E           JSR     PCRLF
07BD 01                 NOP
07BE 01                 NOP

0447                    ORG     $447
0447 BD 18 7B           JSR     ERRLIN     SET FOR NUMBERS
044A 01                 NOP
044B 01                 NOP
044C 01                 NOP

0489                    ORG     $489
0489 BD 18 7B           JSR     ERRLIN
```

```
048C 01                  NOP
048D 01                  NOP
048E 01                  NOP

04D6                     ORG     $4D6
04D6 7E 18 3E            JMP     DOTAP       FIX TAPE ROUTINE

05C1                     ORG     $5C1
05C1 BD 18 82            JSR     PLINE       PRINT LINE OF SOURCE
05C4 01                  NOP

05FA                     ORG     $5FA
05FA BD 18 5D            JSR     PBYTE       DO SPECIAL BYTE

0642                     ORG     $642
0642               PRTSRC EQU     *
0642 7E 18 96            JMP     OUTSRC      DO OUTPUT

0D03                     ORG     $D03
0D03 7E 18 2E            JMP     DOMOD       FIX MODFY

0E87                     ORG     $E87
0E87 BD 18 05            JSR     DOFCC       FIX FCC
0E8A CE 02 00            LDX     #BYTSTK     SET POINTER
0E8D DF 87               STX     BYTPTR      SAVE IT
0E8F DE 96               LDX     OPNPTR      SET TO OPN
0E91 DF 73               STX     XTEMP5      SAVE IT
0E93 BD 13 48            JSR     INDEC       CHECK FOR DECIMAL
0E96 24 59               BCC     TYPE8F      ERROR?
0E98 BD 11 D5            JSR     EVAL        GO EVALUATE

0F42                     ORG     $F42
0F42 BD 18 0B            JSR     DOFCB       SET SPECIAL

10E6                     ORG     $10E6
10E6 7E 18 19            JMP     DOEND       FIX END
10E9 BD 18 12            JSR     DOTTL       FIX TTL & NAM

1059                     ORG     $1059
1059 58                  FCC     'XXX'

11F5                     ORG     $11F5
11F5 BD 18 51            JSR     FIXSTR      FIX #'* PROBLEM

1577                     ORG     $1577

                   * WRITE BINARY TO DISK FILE

1577 DE 6D         MEMCOD LDX    XTEMP2      GET PC
1579 9C 9C               CPX     LSTPCM      CONTIGUOUS CODE?
157B 07                  TPA
157C DE 4B               LDX     PC          GET ACTUAL PC
157E DF 9C               STX     LSTPCM
1580 06                  TAP
```

```
1581 27 12            BEQ     MEM2        IF CONT, SKIP
1583 8D 50            BSR     MEMPU2
1585 CE 19 AC         LDX     #MEMBUF     SET POINTER
1588 96 6D            LDA A   XTEMP2      GET PC
158A A7 01            STA A   1,X         SET IN BUFFER
158C 96 6E            LDA A   XTEMP2+1
158E A7 02            STA A   2,X
1590 CE 19 B0  MEM1   LDX     #MEMBUF+4   GET POINTER
1593 DF 8B            STX     MEMPTR      SAVE POINTER
1595 DE 8B     MEM2   LDX     MEMPTR      GET POINTER
1597 D6 90            LDA B   OPCNT       GET COUNT
1599 96 7E            LDA A   OPCODE      GET BYTE
159B 8D 2D            BSR     MEMPUT      PUT IN BUFFER
159D 5A               DEC B               DEC THE COUNT
159E 27 0B            BEQ     MEM3
15A0 96 7F            LDA A   OP1         GET NEXT
15A2 8D 26            BSR     MEMPUT      PUT IT
15A4 5A               DEC B               DEC THE COUNT
15A5 27 04            BEQ     MEM3
15A7 96 80            LDA A   OP2         GET NEXT BYTE
15A9 8D 1F            BSR     MEMPUT
15AB DF 8B     MEM3   STX     MEMPTR      SAVE POINTER
15AD 96 5A            LDA A   DATFLG      TEST FLAG
15AF 26 01            BNE     EXTDAT
15B1 39        MEM4   RTS                 RETURN

15B2 CE 02 00  EXTDAT LDX     #BYTSTK     SET POINTER
15B5 DF 71            STX     XTEMP4      SAVE IT
15B7 DE 71     MEM5   LDX     XTEMP4      GET POINTER
15B9 9C 87            CPX     BYTPTR      EMPTY?
15BB 27 F4            BEQ     MEM4        FINISHED?
15BD A6 00            LDA A   0,X         GET BYTE
15BF 08               INX                 BUMP POINTER
15C0 DF 71            STX     XTEMP4      SAVE IT
15C2 DE 8B            LDX     MEMPTR      POINT TO BUFFER
15C4 8D 04            BSR     MEMPUT
15C6 DF 8B            STX     MEMPTR      SAVE PLACE
15C8 20 ED            BRA     MEM5        REPEAT

                * PUT BYTE IN BUFFER

15CA A7 00     MEMPUT STA A   0,X         PUT BYTE
15CC 08               INX                 BUMP THE POINTER
15CD 7C 00 9A         INC     MCOUNT      BUMP COUNTER
15D0 8C 1A 74         CPX     #MEMBUF+200
15D3 26 2C            BNE     MEMPU8      FINISHED?
15D5 96 9A     MEMPU2 LDA A   MCOUNT      GET COUNT
15D7 27 28            BEQ     MEMPU8      EMPTY?
15D9 37               PSH B
15DA CE 19 AC         LDX     #MEMBUF     SET POINTER
15DD A7 03            STA A   3,X         SET BYTE COUNT
15DF 16               TAB                 SAVE COUNT
15E0 CB 04            ADD B   #4          ADD IN ADD BYTES
15E2 A6 00     MEMPU4 LDA A   0,X         GET CHARACTER
```

```
15E4 BD 17 F8          JSR     WRITE
15E7 08                INX                 BUMP THE POINTER
15E8 5A                DEC B               DEC COUNT
15E9 26 F7             BNE     MEMPU4
15EB D7 9A             STA B   MCOUNT      CLEAR COUNT
15ED CE 19 AC          LDX     #MEMBUF     RESET POINTER
15F0 DF 8B             STX     MEMPTR
15F2 5F                CLR B
15F3 86 C4             LDA A   #196        SET ADDRESS
15F5 AB 02             ADD A   2,X         FIX ADDRESS
15F7 E9 01             ADC B   1,X
15F9 E7 01             STA B   1,X         SAVE NEW
15FB A7 02             STA A   2,X
15FD 33                PUL B               RESTORE B
15FE CE 19 B0          LDX     #MEMBUF+4
1601 39        MEMPU8  RTS                 RETURN

                       * DISK ASSEMBLER ENTRY

1602 8E A0 7F  DASMB   LDS     #$A07F      SET STACK
1605 CE 17 84          LDX     #DASM95     SET RETURN REG
1608 FF 70 96          STX     RETRNR
160B CE 1B F4          LDX     #BUFBEG     SET SOURCE BUFFER
160E DF 44             STX     SRCBEG
1610 86 FF             LDA A   #$FF
1612 97 28             STA A   OBJFLG
1614 97 29             STA A   PRNFLG
1616 97 2A             STA A   NLFLG
1618 97 2B             STA A   NSFLG
161A 97 2C             STA A   NGFLG
161C 97 2D             STA A   TPFLG
161E 97 33             STA A   LINFLG
1620 97 30             STA A   YFLG
1622 4F                CLR A
1623 97 36             STA A   XFRFLG
1625 97 34             STA A   LINCNT
1627 97 35             STA A   LINCNT+1
1629 97 2E             STA A   END
162B 97 48             STA A   LINBYT      SET LINE BYTE
162D 86 B4             LDA A   #$B4        SET TEST CODE
162F A7 00     DASMB2  STA A   0,X          WRITE CHARACTER
1631 A1 00             CMP A   0,X         CHECK IT
1633 26 06             BNE     DASMB3      LOOP
1635 08                INX                 BUMP TO NEXT
1636 8C 70 00          CPX     #$7000      INTO DOS?
1639 26 F4             BNE     DASMB2      LOOP?
163B 09        DASMB3  DEX
163C DF 42             STX     LBLEND      SET LABLE END
163E 96 42             LDA A   LBLEND      GET HI HALF
1640 D6 43             LDA B   LBLEND+1
1642 81 50             CMP A   #$50        LOTS OF MEMORY?
1644 25 04             BLO     DASM35
1646 80 08             SUB A   #8          SAVE SOME SPACE
```

```
1648 97 42            STA A  LBLEND     SET NEW END
164A D7 41     DASM35 STA B  LBLBEG+1   SET POINTER
164C 37               PSH B
164D 16               TAB
164E C4 F0            AND B  #$F0       MASK OFF
1650 C1 60            CMP B  #$60
1652 25 02            BLO    DASM37
1654 C0 10            SUB B  #$10       REMOVE SOME
1656 C0 10     DASM37 SUB B  #$10       FIX VALUE
1658 54               LSR B
1659 10               SBA
165A 97 40            STA A  LBLBEG     SET BEGINNING
165C 33               PUL B              RESTORE B
165D 7C 00 41         INC    LBLBEG+1
1660 26 03            BNE    DASMB4
1662 7C 00 40         INC    LBLBEG
1665 4A        DASMB4 DEC A              SET VALUE
1666 97 3B            STA A  BUFEND     SAVE BUFFER
1668 D7 3C            STA B  BUFEND+1
166A CE 1A 74         LDX    #RFCB      POINT TO FCB
166D BD 71 27         JSR    GETFIL     GET FILE NAME
1670 25 35            BCS    DASM47
1672 CE 1A 74         LDX    #RFCB      POINT TO FCB
1675 86 01            LDA A  #1         SET EXT
1677 BD 71 2D         JSR    SETEXT
167A C6 0C            LDA B  #12        SET COUNTER
167C CE 1A 77         LDX    #RFCB+3    SET POINTER
167F A6 00     DASM45 LDA A  0,X        GET CHARACTER
1681 A7 C0            STA A  192,X      MOVE TO WFCB
1683 08               INX                BUMP TO NEXT
1684 5A               DEC B              DEC THE COUNT
1685 26 F8            BNE    DASM45
1687 CE 1B 34         LDX    #WFCB      SET POINTER
168A 6F 0C            CLR    12,X       CLEAR EXT
168C CE 1A 74         LDX    #RFCB      SET POINTER
168F 86 01            LDA A  #1         OPEN FOR READ
1691 A7 00            STA A  0,X
1693 BD 78 06         JSR    FMS        CALL FMS
1696 26 5C            BNE    DASM72     ERROR?
1698 BD 17 8A         JSR    TSTTRM     TEST TERM
169B 27 39            BEQ    DASMB7
169D 2B 18            BMI    DASMB6
169F CE 1B 34         LDX    #WFCB      SET TO FCB
16A2 BD 71 27         JSR    GETFIL     GET FILE NAME
16A5 24 09            BCC    DASMB5     ERROR?
16A7 CE 19 73  DASM47 LDX    #ILST      POINT TO STRING
16AA BD 71 18  DASM48 JSR    PSTRNG     OUTPUT IT
16AD 7E 71 03         JMP    WARMS      RETURN TO DOS
16B0 BD 17 8A  DASMB5 JSR    TSTTRM     TEST TERM
16B3 27 21            BEQ    DASMB7
16B5 2A 1F            BPL    DASMB7
16B7 BD 71 21  DASMB6 JSR    NXTCH      GET CHARACTER
16BA 25 1A            BCS    DASMB7
16BC CE 19 44         LDX    #TABLE     SET POINTER
```

```
16BF A1 00      DASM62  CMP A   0,X      CHECK CHAR
16C1 27 0C              BEQ     DASM67
16C3 08                 INX             BUMP THE POINTER
16C4 08                 INX
16C5 08                 INX
16C6 6D 00              TST     0,X      END OF TABLE?
16C8 26 F5              BNE     DASM62
16CA CE 19 5A   DASM65  LDX     #IOST    POINT TO STRING
16CD 20 DB              BRA     DASM48
16CF 08         DASM67  INX             BUMP THE POINTER
16D0 EE 00              LDX     0,X      GET ADDRESS
16D2 6F 00              CLR     0,X      SET COND
16D4 20 E1              BRA     DASMB6   REPEAT
16D6 96 28      DASMB7  LDA A   OBJFLG   GET FLAG
16D8 27 5C              BEQ     DASMB8   BINARY?
16DA CE 1B 34           LDX     #WFCB    SET TO FCB
16DD 86 00              LDA A   #0
16DF BD 71 2D           JSR     SETEXT   SET EXTENSION
16E2 CE 1B 34           LDX     #WFCB
16E5 86 02              LDA A   #2       SET FOR WRITE
16E7 A7 00              STA A   0,X
16E9 BD 78 06           JSR     FMS      OPEN FILE
16EC 26 06              BNE     DASM72
16EE 86 FF              LDA A   #$FF     SET SPC COMP
16F0 A7 3B              STA A   59,X
16F2 20 42              BRA     DASMB8
16F4 A6 01      DASM72  LDA A   1,X      GET ERROR
16F6 81 03              CMP A   #FEER    FILE EXISTS?
16F8 26 29              BNE     DASM74
16FA 96 30              LDA A   YFLG     CHECK FOR Y
16FC 27 13              BEQ     DASM73
16FE 7C 70 A3           INC     OUTCHN   SET CHANNEL
1701 CE 19 92           LDX     #DLST    POINT TO STRING
1704 BD 71 18           JSR     PSTRNG   OUTPUT IT
1707 BD 71 0F           JSR     GETCHR   GET RESPONSE
170A 7F 70 A3           CLR     OUTCHN   RESET CHAN
170D 81 59              CMP A   #'Y      IS IT YES?
170F 26 1F              BNE     DASM77   EXIT
1711 CE 1B 34   DASM73  LDX     #WFCB    POINT TO FCB
1714 86 0C              LDA A   #12      SET FOR DELETE
1716 A7 00              STA A   0,X
1718 BD 78 06           JSR     FMS      CALL FMS
171B 26 10              BNE     DASM75   ERROR?
171D A6 24              LDA A   36,X     RESTORE NAME
171F A7 04              STA A   4,X
1721 20 B3              BRA     DASMB7
1723 81 04      DASM74  CMP A   #NFER     NO FILE?
1725 26 06              BNE     DASM75
1727 CE 19 85           LDX     #NFST    POINT TO STRING
172A 7E 16 AA           JMP     DASM48
172D BD 71 3C   DASM75  JSR     RPTERR   REPORT ERROR
1730 BD 78 03   DASM77  JSR     FMSCLS
1733 7E 71 03           JMP     WARMS    DO DOS
1736 BD 03 26   DASMB8  JSR     P1INIT
```

```
1739 BD 17 AF  DASM86  JSR     FILBUF      FILL BUFFER
173C BD 03 B1          JSR     PASONE      DO PASS
173F 96 2E             LDA A   END         CHECK END
1741 27 F6             BEQ     DASM86
1743 7F 00 2E          CLR     END         CLEAR FLAG
1746 CE 1A 74          LDX     #RFCB       POINT TO FCB
1749 86 05             LDA A   #5          REWIND FILE
174B A7 00             STA A   0,X
174D BD 78 06          JSR     FMS         CALL FMS
1750 26 DB             BNE     DASM75
1752 BD 03 6F          JSR     P2INIT      DO INITLZ
1755 5F                CLR B               CLEAR FLAG
1756 D7 9A             STA B   MCOUNT
1758 D7 32             STA B   SPCFLG      CLEAR SPECIALS
175A D7 31             STA B   FCBFLG
175C 86 02             LDA A   #2          SET BOF BYTE
175E B7 19 AC          STA A   MEMBUF
1761 CE 19 B0          LDX     #MEMBUF+4   SET POINTER
1764 DF 8B             STX     MEMPTR
1766 96 28             LDA A   OBJFLG      SET ALL OPTIONS
1768 97 B3             STA A   MEMORY      BINARY
176A 96 2A             LDA A   NLFLG       LIST
176C 97 AE             STA A   LIST
176E 96 2B             LDA A   NSFLG       SYMBOLS
1770 97 AF             STA A   SYMBOL
1772 96 2C             LDA A   NGFLG       GEN
1774 97 B0             STA A   GENER
1776 96 2D             LDA A   TPFLG       TAPE
1778 43                COM A
1779 97 B2             STA A   TAPE
177B 8D 32     DASMB9  BSR     FILBUF      FILL BUFFER
177D BD 03 D9          JSR     PASTWO      DO PASS
1780 96 2E             LDA A   END         FINISHED?
1782 27 F7             BEQ     DASMB9
1784 BD 78 03  DASM95  JSR     FMSCLS      CLOSE FILES
1787 7E 71 03          JMP     WARMS       GOTO DOS


               * TEST LAST TERM

178A B6 70 91  TSTTRM  LDA A   LSTTRM      GET TERM
178D 81 0D             CMP A   #$D         IS IT CR?
178F 27 1D             BEQ     TSTTR4
1791 B1 70 82          CMP A   DEOL        IS IT EOL?
1794 27 18             BEQ     TSTTR4
1796 81 2B             CMP A   #'+         IS IT MINUS
1798 26 03             BNE     TSTTR2
179A 86 FF     TSTTR1  LDA A   #$FF        SET NEG
179C 39                RTS                 RETURN
179D FE 70 94  TSTTR2  LDX     BUFPNT      SET TO BUFFER
17A0 A6 00             LDA A   0,X         GET CHARACTER
17A2 08                INX                 BUMP THE POINTER
17A3 81 2B             CMP A   #'+         IS IT '+'?
17A5 26 05             BNE     TSTTR3
```

```
17A7 FF 70 94            STX     BUFPNT     SET NEW POINTER
17AA 20 EE               BRA     TSTTR1
17AC 86 01      TSTTR3   LDA A   #1         SET PLUS
17AE 39         TSTTR4   RTS                RETURN

                * FILL RAM BUFFER WITH SOURCE

17AF 7F 00 2F   FILBUF   CLR     FLG        CLEAR FLAG
17B2 DE 44               LDX     SRCBEG     SET POINTER
17B4 DF 3D      FILBU2   STX     INDEX5     SAVE POINTER
17B6 CE 1A 74   FILB22   LDX     #RFCB      POINT TO FCB
17B9 BD 78 06            JSR     FMS        GET CHARACTER
17BC 27 0B               BEQ     FILBU3
17BE A6 01               LDA A   1,X        CHECK ERROR
17C0 81 08               CMP A   #8         IS IT EOF?
17C2 27 03               BEQ     FILB25
17C4 7E 17 2D   FILB23   JMP     DASM75     REPORT ERROR
17C7 86 1A      FILB25   LDA A   #$1A       SET EOF
17C9 DE 3D      FILBU3   LDX     INDEX5     RESTORE POINTER
17CB 9C 3B               CPX     BUFEND     END OF BUFFER?
17CD 26 03               BNE     FILBU4
17CF 7C 00 2F            INC     FLG        SET FLAG
17D2 81 0D      FILBU4   CMP A   #$D        IS IT CR?
17D4 27 18               BEQ     FILBU7
17D6 81 1A               CMP A   #$1A       IS IT EOF?
17D8 26 0B               BNE     FILBU5
17DA 97 2E               STA A   END        SET END
17DC 09         FILB45   DEX                DEC THE POINTER
17DD A6 00               LDA A   0,X        CHECK CHARACTER
17DF 81 0D               CMP A   #$D        IS IT CR?
17E1 26 F9               BNE     FILB45
17E3 20 10               BRA     FILBU8
17E5 81 1F      FILBU5   CMP A   #$1F       IS IT CNTRL?
17E7 23 CB               BLS     FILBU2
17E9 A7 00      FILB55   STA A   0,X        PUT CHARACTER
17EB 08         FILBU6   INX                BUMP THE POINTER
17EC 20 C6               BRA     FILBU2     REPEAT
17EE A7 00      FILBU7   STA A   0,X        PUT CR
17F0 7D 00 2F            TST     FLG        FINISHED?
17F3 27 F6               BEQ     FILBU6
17F5 DF 46      FILBU8   STX     SRCEND     SET POINTER
17F7 39                  RTS                RETURN

                * WRITE TO FILE

17F8 DF 3D      WRITE    STX     INDEX5     SAVE POINTER
17FA CE 1B 34            LDX     #WFCB      SET TO FCB
17FD BD 78 06            JSR     FMS        PUT CHARACTER
1800 26 C2               BNE     FILB23
1802 DE 3D      WRITE2   LDX     INDEX5     RESTORE
1804 39                  RTS                RETURN
```

```
                    * SPECIAL PATCHES AND MODS

1805 86 FF      DOFCC   LDA A   #$FF         SET ERROR FLAG
1807 97 56              STA A   ERRFLG
1809 20 0A              BRA     DOTTL2

180B CE 02 00   DOFCB   LDX     #BYTSTK     SET POINTER
180E 7C 00 31           INC     FCBFLG      SET FLAG
1811 39                 RTS                 RETURN

1812 7F 00 59   DOTTL   CLR     PCFLG
1815 7C 00 32   DOTTL2  INC     SPCFLG      SET SPECIAL FLAG
1818 39                 RTS

                    * SET TRANSFER ADDRESS

1819 97 58      DOEND   STA A   $58         SET END
181B DE 94              LDX     OPTPTR      SET POINTER
181D A6 03              LDA A   3,X         FIELD THERE?
181F 81 0D              CMP A   #$D         IS IT CR?
1821 27 0A              BEQ     DOEND1
1823 BD 11 D5           JSR     EVAL        EVALUATE FIELD
1826 DE 7B              LDX     QTEMP       GET VALUE
1828 DF 37              STX     XFRADR      SET ADDRESS
182A 7C 00 36           INC     XFRFLG      SET FLAG
182D 39         DOEND1  RTS

182E 7F 00 AB   DOMOD   CLR     MODFY       FIX FLAG
1831 7E 0C 86           JMP     ADDPC1      FINISH UP

                    * FIX DATA FLAG IN FCB

1834 7F 00 5A   FIXDAT  CLR     DATFLG      CLEAR OUT
1837 96 B0              LDA A   GENER       CHECK IF GENERATION
1839 27 F2              BEQ     DOEND1
183B 7E 05 D3           JMP     $5D3        CONTINUE

183E 96 2D      DOTAP   LDA A   TPFLG
1840 26 03              BNE     DOTAP2
1842 7E 04 F1           JMP     $4F1        FINISH TAPE OUT
1845 7E 04 DA   DOTAP2  JMP     $4DA        CONTINUE

1848 BD 11 31   DOSYM   JSR     $1131       EJECT TO NEW PAGE
184B BD 71 1E           JSR     PCRLF       OUTPUT CR & LF
184E 7E 71 1E           JMP     PCRLF       RETURN

                    * FIX #'* PROBLEM

1851 09         FIXSTR  DEX                 DEC THE POINTER
1852 A6 00              LDA A   0,X
1854 08                 INX                 BUMP TO NEXT
1855 81 27              CMP A   #''         IS IT QUOTE?
1857 27 02              BEQ     FIXST2
1859 9C 96              CPX     OPNPTR      CHECK POINTER
```

```
185B 07          FIXST2  TPA             SAVE STATUS
185C 39                  RTS             RETURN

                 * SPECAIL BYTE OUTPUT

185D BD 07 BA    PBYTE   JSR     $7BA    OUTPUT CR & LF
1860 7F 00 5A            CLR     DATFLG
1863 7D 00 33            TST     LINFLG  NUMBERS?
1866 26 06               BNE     PBYTE4
1868 BD 0C C5            JSR     $CC5    OUT 3 SPACES
186B BD 0C C5            JSR     $CC5    OUT 3 MORE SPACES
186E 7E 06 02    PBYTE4  JMP     PRTDAT+3 RETURN

1871 DE 4B       P2AUX   LDX     PC      SETUP NEW PC
1873 DF 6D               STX     XTEMP2  SAVE VALUE
1875 DE 34               LDX     LINCNT  GET COUNT
1877 08                  INX             BUMP BY ONE
1878 DF 34               STX     LINCNT  SAVE NEW
187A 39                  RTS             RETURN

187B BD 07 BA    ERRLIN  JSR     $7BA    DO CR & LF
187E C6 01               LDA B   #1
1880 20 08               BRA     PLINE1

                 * LINE NUMBER OUTPUT

1882 BD 07 BA    PLINE   JSR     $7BA    OUTPUT CR & LF
1885 D6 33               LDA B   LINFLG  CHECK FOR NUMBERS
1887 26 0A               BNE     PLINE2
1889 5C                  INC B           SET FOR SPACES
188A CE 00 34    PLINE1  LDX     #LINCNT SET POINTER
188D BD 71 33            JSR     OUTDEC  OUTPUT NUMBER
1890 BD 03 1E            JSR     $31E    OUTPUT A SPACE
1893 BD 06 02    PLINE2  JSR     PRTDAT+3 DO DATA

                 * SPECIAL FIELDS OUTPUT ROUTINE

008D             LINPTR  EQU     $8D

1896 DE 8D       OUTSRC  LDX     LINPTR
1898 5F                  CLR B
1899 A6 00               LDA A   0,X
189B 81 2A               CMP A   #$2A    CHECK IF COMMENT
189D 27 5D               BEQ     FINOUT
189F 81 20       OUTSR1  CMP A   #$20    IS IT SPACE?
18A1 27 02               BEQ     OUTSR2
18A3 8D 69               BSR     OUTFLD  OUTPUT NEXT FIELD
18A5 86 20       OUTSR2  LDA A   #$20
18A7 BD 71 12            JSR     PUTCHR  OUTPUT SPACE
18AA 5C                  INC B           BUMP COUNTER
18AB C1 08               CMP B   #8      COLUMN 9 YET?
18AD 25 F6               BLO     OUTSR2
18AF 8D 6E       OUTSR3  BSR     NEXTC   GET NEXT CHARACTER
18B1 BD 71 12            JSR     PUTCHR  OUTPUT IT
```

```
18B4 5C                  INC B           BUMP COUNT
18B5 C1 0B               CMP B  #11      COLUMN 12?
18B7 25 F6               BLO    OUTSR3
18B9 7D 00 AB            TST    MODFY    A OR B REF?
18BC 27 0C               BEQ    OUTSR4
18BE 86 20               LDA A  #$20     SETUP SPACE
18C0 BD 71 12            JSR    PUTCHR   OUTPUT IT
18C3 5C                  INC B           BUMP COUNT
18C4 8D 59               BSR    NEXTC    GET NEXT CHAR
18C6 BD 71 12            JSR    PUTCHR   OUTPUT IT
18C9 5C                  INC B           BUMP COUNT
18CA 7D 00 32  OUTSR4    TST    SPCFLG   FLAG SET?
18CD 26 0F               BNE    OUTSR5   IF SO, FINISH
18CF 7D 00 31            TST    FCBFLG   FCB ?
18D2 26 0A               BNE    OUTSR5
18D4 96 90               LDA A  OPCNT    GET OP COUNT
18D6 81 01               CMP A  #1       IS IT 1 BYTE?
18D8 26 04               BNE    OUTSR5
18DA 8D 43               BSR    NEXTC    GET NEXT CHAR
18DC 20 14               BRA    OUTSR6
18DE 86 20     OUTSR5    LDA A  #$20     SET UP SPACE
18E0 BD 71 12            JSR    PUTCHR   OUTPUT IT
18E3 5C                  INC B           BUMP COL COUNT
18E4 C1 0F               CMP B  #15      COLUMN 16 YET?
18E6 25 F6               BLO    OUTSR5   REPEAT
18E8 8D 35               BSR    NEXTC    GET NEXT
18EA 96 32               LDA A  SPCFLG   CHECK SPECIALS
18EC 26 0E               BNE    FINOUT   FINISH?
18EE 8D 1E               BSR    OUTFLD   OUTPUT NEXT FIELD
18F0 8D 2D               BSR    NEXTC    GET TO NEXT
18F2 86 20     OUTSR6    LDA A  #$20     SET SPACE
18F4 BD 71 12            JSR    PUTCHR   OUTPUT IT
18F7 5C                  INC B           BUMP COUNT
18F8 C1 19               CMP B  #25      COLUMN 26?
18FA 25 F6               BLO    OUTSR6   REPEAT

18FC 7F 00 32  FINOUT    CLR    SPCFLG   CLEAR FLAG
18FF 7F 00 31            CLR    FCBFLG   CLEAR FCB
1902 A6 00               LDA A  0,X      GET NEXT CHARACTER
1904 81 0D               CMP A  #$D      IS IT CR?
1906 27 24               BEQ    NEXTC4
1908 BD 71 12            JSR    PUTCHR   OUTPUT IT
190B 08                  INX             BUMP TO NEXT
190C 20 EE               BRA    FINOUT

                * OUTPUT FIELD

190E A6 00     OUTFLD    LDA A  0,X      GET CHARACTER
1910 81 0D               CMP A  #$D      IS IT CR?
1912 27 16               BEQ    NEXTC2
1914 81 20               CMP A  #$20     IS IT SPACE?
1916 27 14               BEQ    NEXTC4
1918 BD 71 12            JSR    PUTCHR   OUTPUT CHARACTER
191B 08                  INX             BUMP TO NEXT
```

```
191C 5C                 INC B               BUMP COUNT
191D 20 EF              BRA     OUTFLD

                * GET NEXT CHAR SKIPPING SPACES

191F 08         NEXTC   INX                 BUMP POINTER
1920 A6 00              LDA A   0,X         GET CHARACTER
1922 81 20              CMP A   #$20        IS IT SPACE?
1924 27 F9              BEQ     NEXTC
1926 81 0D              CMP A   #$D         IS IT CR?
1928 26 02              BNE     NEXTC4
192A 33         NEXTC2  PUL B               RESTORE STACK
192B 33                 PUL B
192C 39         NEXTC4  RTS                 RETURN

                * FINISH OBJECT DUMP

192D BD 15 D5   FINOBJ  JSR     MEMPU2      DUMP BUFFER
1930 96 36              LDA A   XFRFLG      TEST FLAG
1932 27 F8              BEQ     NEXTC4      XFR ADDRESS?
1934 86 16              LDA A   #$16        SET RECORD
1936 BD 17 F8           JSR     WRITE       WRITE TO DISK
1939 96 37              LDA A   XFRADR      GET ADDRESS
193B BD 17 F8           JSR     WRITE       OUTPUT IT
193E 96 38              LDA A   XFRADR+1
1940 BD 17 F8           JSR     WRITE       FINISH UP
1943 39                 RTS                 RETURN

                * OPTION TABLE

1944 42         TABLE   FCC     'B'
1945 00 28              FDB     OBJFLG
1947 4C                 FCC     'L'
1948 00 2A              FDB     NLFLG
194A 53                 FCC     'S'
194B 00 2B              FDB     NSFLG
194D 47                 FCC     'G'
194E 00 2C              FDB     NGFLG
1950 54                 FCC     'T'
1951 00 2D              FDB     TPFLG
1953 4E                 FCC     'N'
1954 00 33              FDB     LINFLG
1956 59                 FCC     'Y'
1957 00 30              FDB     YFLG
1959 00                 FCB     0

                * STRINGS

195A 49         IOST    FCC     'ILLEGAL OPTION SPECIFIED'
1972 04                 FCB     4
1973 49         ILST    FCC     'ILLEGAL FILE NAME'
1984 04                 FCB     4
1985 4E         NFST    FCC     'NO SUCH FILE'
1991 04                 FCB     4
```

```
 1992 44          DLST     FCC     'DELETE OLD BINARY (Y-N)? '
 19AB 04                   FCB     4

 19AC             MEMBUF   RMB     200

 1A74             RFCB     RMB     192
 1B34             WFCB     RMB     192

 1BF4             BUFBEG   EQU     *

                           END
```

NO ERROR(S) DETECTED

SYMBOL TABLE:

ADDPC1	0C86	ASN	708C	BUFBEG	1BF4	BUFEND	003B	BUFPNT	7094
BYTPTR	0087	BYTSTK	0200	DASM35	164A	DASM37	1656	DASM45	167F
DASM47	16A7	DASM48	16AA	DASM62	16BF	DASM65	16CA	DASM67	16CF
DASM72	16F4	DASM73	1711	DASM74	1723	DASM75	172D	DASM77	1730
DASM86	1739	DASM95	1784	DASMB	1602	DASMB2	162F	DASMB3	163B
DASMB4	1665	DASMB5	16B0	DASMB6	16B7	DASMB7	16D6	DASMB8	1736
DASMB9	177B	DATFLG	005A	DEOL	7082	DLST	1992	DOEND	1819
DOEND1	182D	DOFCB	180B	DOFCC	1805	DOMOD	182E	DOSYM	1848
DOTAP	183E	DOTAP2	1845	DOTTL	1812	DOTTL2	1815	END	002E
ERRFLG	0056	ERRLIN	187B	EVAL	11D5	EXTDAT	15B2	FCB	7740
FCBFLG	0031	FEER	0003	FILB22	17B6	FILB23	17C4	FILB25	17C7
FILB45	17DC	FILB55	17E9	FILBU2	17B4	FILBU3	17C9	FILBU4	17D2
FILBU5	17E5	FILBU6	17EB	FILBU7	17EE	FILBU8	17F5	FILBUF	17AF
FINOBJ	192D	FINOUT	18FC	FIXDAT	1834	FIXST2	185B	FIXSTR	1851
FLG	002F	FMS	7806	FMSCLS	7803	GENER	00B0	GETCHR	710F
GETFIL	7127	GETHEX	713F	ILST	1973	INBUF	7115	INDEC	1348
INDEX5	003D	IOST	195A	LBLBEG	0040	LBLEND	0042	LINBYT	0048
LINCNT	0034	LINFLG	0033	LINPTR	008D	LIST	00AE	LSTPCM	009C
LSTTRM	7091	MCOUNT	009A	MEM1	1590	MEM2	1595	MEM3	15AB
MEM4	15B1	MEM5	15B7	MEMBUF	19AC	MEMCOD	1577	MEMORY	00B3
MEMPTR	008B	MEMPU2	15D5	MEMPU4	15E2	MEMPU8	1601	MEMPUT	15CA
MODFY	00AB	NEXTC	191F	NEXTC2	192A	NEXTC4	192C	NFER	0004
NFST	1985	NGFLG	002C	NLFLG	002A	NSFLG	002B	NXTCH	7121
OBJFLG	0028	OP1	007F	OP2	0080	OPCNT	0090	OPCODE	007E
OPNPTR	0096	OPTPTR	0094	OUT2S	0CC7	OUTCHN	70A3	OUTDEC	7133
OUTFLD	190E	OUTHEX	7139	OUTSR1	189F	OUTSR2	18A5	OUTSR3	18AF
OUTSR4	18CA	OUTSR5	18DE	OUTSR6	18F2	OUTSRC	1896	P1INIT	0326
P2AUX	1871	P2INIT	036F	PASONE	03B1	PASTWO	03D9	PBYTE	185D
PBYTE4	186E	PC	004B	PCFLG	0059	PCRLF	711E	PLINE	1882
PLINE1	188A	PLINE2	1893	PRNFLG	0029	PRTDAT	05FF	PRTSRC	0642
PSTRNG	7118	PUTCHR	7112	QTEMP	007B	RETRNR	7096	RFCB	1A74
RPTERR	713C	SETEXT	712D	SPC	0020	SPCFLG	0032	SRCBEG	0044
SRCEND	0046	START	0300	SYMBOL	00AF	TABLE	1944	TAPE	00B2
TMPOUT	0039	TPFLG	002D	TSTTR1	179A	TSTTR2	179D	TSTTR3	17AC
TSTTR4	17AE	TSTTRM	178A	TYPE8F	0EF1	WARMS	7103	WFCB	1B34
WRITE	17F8	WRITE2	1802	XFRADR	0037	XFRFLG	0036	XTEMP2	006D
XTEMP4	0071	XTEMP5	0073	YFLG	0030				

ARE SOFTWARE SOFTW
SOFTWARE SOFTWARI
TWARE SOFTWARE SO
ARE SOFTWARE SOFTW
SOFTWARE SOFTWARI
TWARE SOFTWARE SO
RE SOFTWARE SOFTW
SOFTWARE SOFTWARI
TWARE SOFTWARE SO
ARE SOFTWARE SOFTW
SOFTWARE SOFTWARI
TWARE SOFTWARE SO
RE SOFTWARE SOFTW
SOFTWARE SOFTWARI
TWARE SOFTWARE SO
ARE SOFTWARE SOFTW
SOFTWARE SOFTWARI